Modélisation et conception de schémas de signatures et de protocoles de paiements anonymes

Thèse de doctorat de l'Université Clermont Auvergne
Spécialité: Informatique

École Doctorale Science Pour l'Ingénieur (SPI)
Laboratoire d'Informatique, de Modélisation et d'Optimisation des Systèmes (LIMOS)
CNRS, UMR 6158, LIMOS, Aubière, France

Thèse soutenu le 13 décembre 2024 par

Charles Olivier--Anclin

Composition du jury:

Olivier Blazy Professeur des universités, Ecole Polytechnique	President du jury
Sébastien Canard Professeur des universités, Télécom Paris	Rapporteur
Cristina Onete Maître de conférences HDR, XLIM, Université de Limoges	Rapporteuse
Katharina Boudgoust Chargée de Recherche, CNRS, Université de Montpellier, LIRMM	Examinatrice
Dario Fiore Associate Research Professor, IMDEA Software électronique Institute	Examinateur
David Pointcheval Directeur de Recherche, ENS, CNRS, en détachement chez Cosmian	Examinateur
Pascal Lafourcade Professeur des universités, LIMOS, Université Clermont Auvergne	Directeur de Thèse
Xavier Bultel Maître de conférences, INSA CVL, LIFO, INRIA, Université d'Orléans	Encadrant de Thèse

Résumé

L'anonymat est une propriété de sécurité d'une primitive cryptographique ou d'un protocole de communication qui élimine ou limite l'identification des utilisateurs. Elle a suscité un intérêt prédominant et a évolué pour devenir un élément central des enjeux sociétaux.

Dans cette thèse, nous contribuons à rendre l'anonymat plus accessible. À cette fin, nous étudions des signatures électroniques avec différentes propriétés d'anonymat, afin de développer les mécanismes qui permettront de restreindre le degré d'identification aux cas strictement nécessaires. Plutôt que d'imposer une identification complète dans des situations où cela ne serait pas nécessaire. Nous avons porté notre attention sur l'analyse des propriétés d'anonymat des signatures d'anneaux liées, qui assurent l'anonymat tout en maintenant un lien entre les signatures des mêmes signataires, ce qui est particulièrement pertinent dans le contexte du vote électronique. Notre intérêt s'est aussi porté sur les mécanismes de délégation de signatures. Nous proposons une méthode permettant d'assurer l'anonymat du délégué tout en limitant le nombre de signatures qu'il peut émettre. Le fait de dépasser le seuil implique la révélation de son identité, ainsi que la possibilité de retracer toutes les signatures qu'il aura émises. Dans ces deux études, nous avons adopté le paradigme de la cryptographie prouvée et nous nous appuierons sur des modèles dits calculatoires.

Considérant également un protocole largement déployé - le protocole de paiement par carte EMV - nous proposons des constructions offrant différents niveaux d'anonymat et compatible avec la norme existante. Nous montrons ici que, malgré les multiples contraintes normatives et les lois imposant une identification permanente des parties, il est possible de concevoir des architectures plus respectueuses de la vie privée qui peut s'intègre directement au système actuel. Cela vise à démontrer que la protection de l'identité des entités reste toujours possible même dans un protocole établi de longue date.

Ainsi, notre étude ne se limite pas à l'anonymat ; nous nous penchons également sur d'autres propriétés essentielles, pour modérer un anonymat trop fort et assurer que les primitives soient facilement intégrables dans des systèmes concrets.

Abstract

Cryptographic anonymity is a security property of cryptographic primitives or communication protocols eliminating or limiting user identification. It has been of predominant interest and has now evolved to become central to societal issues.

In this thesis, we contribute to making anonymity more attainable. With this in mind, we study electronic signatures with various anonymity properties, in order to develop mechanisms that will make it possible, to restrict the degree of identification to what is strictly necessary. This, rather than imposing full identification in situations where it should not be required. We propose the study of the anonymity properties of linkable ring signatures. These signatures schemes ensure anonymity while maintaining a link between the signatures of the same signer, a property that is particularly relevant in the context of electronic voting, for example. We are also interested in signature delegation mechanisms. Furthermore, we propose a method for ensuring the anonymity of the delegate while limiting the number of signatures that can be issued. Exceeding the threshold implies revealing the identity of the signer, as well as the possibility of tracing all its issued signatures. In these two studies, we have adopted the paradigm of proven cryptography and will rely on so-called computational models.

Also considering a more practical protocol - the EMV card payment protocol - we propose constructions allowing degrees of anonymity that comply with the current standard. We show here that, despite the many normative constraints and laws imposing permanent identification of parties, it is possible to design more privacy-friendly architectures that can be integrated directly into the current system. The aim is to demonstrate that it is still possible to protect the identity of entities even in a long-established protocol.

Thus, our study is not limited to anonymity; we also look at other essential properties to moderate too strong anonymity and ensure that primitives can be easily integrated into concrete schemes.

Contents

Chapter 1

Introduction

1.1 Cryptology

Throughout the development of cryptography, the word *privacy* has had multiple predominant meanings. It first sense was associated to the prevention of unauthorised extraction of information from a communication made over an unsecured channel. Originally, *privacy of messages* was ensured by symmetrical encryption schemes, in which the sender and recipient share a common secret, called *secret keys* and used to encrypt the message. This was at a time when the field was still in its early foundations and encryption relied solely on *secret symmetric keys* that needed to be shared between the sender and the recipient and, above all, no one else. Numerous methods were used to achieve shared secrecy of symmetric keys, including face-to-face meetings and relying on trusted third parties. However, the secret had to pass from the sender to the recipient without being disclosed and before any private communication could take place. This is a major inconvenient, because to send a secure message between two entities, another message (in this case, the key) had to be transmitted securely from one entity to another. Hence, as noted by Diffie and Hellman [DH76] in 1976,

> "Cryptography was unable to meet the (*security*) requirements, in that its use would impose such severe inconveniences on the system users, as to eliminate many of the benefits."

This is particularly relevant to today's modern global networks, where the requirement is unrealistic given the billions of new device connections needed every day.

Acknowledging the need to solve this key-exchange limitation, Diffie and Hellman have set "*New Directions in Cryptography*" [DH76]. Their seminal work has defined the modern foundation of cryptography and pushed a new concept called *public key cryptography*. Amongst this concept asymmetric encryption assumes that the recipient has a key pair for which one is private and the other public. In this case, the sender uses the recipient's public key to encrypt the message, while the recipient uses its private key to decrypt it. Besides this major difference, the two types of encryption must offer the same functionality and the same level of privacy protection.

With the advent of public key cryptography, the notion of privacy took on a broader meaning: it was no longer just a question of guaranteeing the non-disclosure of encrypted messages (ciphertexts), the identity of recipient users or devices needed to be verified. How can you be sure of the identity of the owner of a public key without meeting him? A

cryptographic mechanism enabling authentication of the recipient was therefore deemed necessary for secure communications. Otherwise, the entity would always have to rely on pre-shared keys (which cannot be completely eliminated in reality).

Luckily, while introducing the first asymmetric encryption scheme, Rivest, Shamir and Adelman [RSA78] also introduced the first *digital signature scheme*. A digital signature is a cryptographic mechanism also based on a pair of keys, that is in today's world the direct counterpart of the pen and paper signatures we all know about. To provide more intuition on the concept we illustrate it in Figure 1.1 by representing the signature, like a seal (affixed on a letter) *i.e.,* one user producing a signature on its own. The red dot there represents the signature made with a public key leading to "red" signatures, which is considered to be unique, *i.e.,* any other public key would result in a signature of a different colour. Digital signatures can be used to verify the authenticity and integrity of digital messages or documents, guaranteeing that the identity of the sender is confirmed and that the content has not been altered during transmission or storage. They are expected to be unforgeable, *i.e.,* they cannot be produced easily without knowing the secret key. Today, digital signatures play a crucial role in protecting against frauds, forgeries and data breaches. They also allow to authenticate the owner of a public key under some trust assumptions thus deporting it to other and less entities that may be easier and more secure to communicate with.

Figure 1.1: Seal of a Digital Signature[1]. (see Definition 13)

However, the implementation of authentication mechanisms can raise significant privacy issues, as identities can be considered as sensitive information. In many applications, it is not always necessary to authenticate a specific user. Take the example of a read-only document shared by several people and stored on a server. Is it necessary to authenticate the person accessing the document, or is it sufficient to prevent unauthorised access? In many cases where the document is not sensitive, preventing unauthorised access may be sufficient. In these circumstances, it is often enough to prove the right to access the document without revealing the identity of the user. Although it may seem surprising, this can be achieved through the use of some digital signature schemes.

This type of privacy-preserving authentication mechanism is precisely what we explore in this thesis, they fall within the field of modern cryptographic primitives and we focus specifically on the privacy of entities. This concept, in cryptography, is called anonymity. Our primary objectives and contributions involve the study of signature-based authentication methods, the formalisation of their security, and the development of practical and efficient means to protect privacy. We provide means to prevent unnecessary or unwanted disclosure of the identities in some specific use scenarios. To ensure the applicability and practicality of our primitives (signature schemes in this thesis) we consider the need to limit the anonymity in most application domains. Total

[1]This schematic representations and the upcoming ones are inspired from the presentation of [AHAN+22] by Elena Pagnin in PKC '22.

anonymity, regardless of the circumstances, can lead to fraud or reprehensible behaviour going unpunished. Hence, we conside the need of requiring forms of *linkability* of two authentication, *traceability* of the user in case of fraud or even in the stronger scenario full *auditability* of the system, as required by the banking industry amongst others.

1.2 Authentication and Anonymity

In today's systems, anonymity often comes into conflict with utility: users who divulge minimal information about themselves are faced with functional restrictions, and service providers may view them with suspicion. This holds despite the strongly regulated treatments of personal data imposed by the *General Data Protection Regulation* (GDPR) [EU16] in the European Union. One of its requirement is the minimal collection and treatment of personal data for each application. However it has appeared that quantifying the amount of data required by one given application is no easy task. Take for example the balance between the usage of cash and bankcard payments.

One is (almost) fully anonymous (the cash), while bankcard payments do not guarantee any form of anonymity for the customer against the merchant, the bank, or even eavesdropper listening around the point of sale. Research has focused on advanced authentication mechanisms since their apearance. The introduction of advanced properties into signature scheme for authentication purposes began in 1982 with Chaum's *blind signature scheme* [Cha83a] which was originally intended for anonymous card payments.

This type of signatures appears as a seal affixed on a letter, similarly to a signature represented in Figure 1.1. The difference lies in how it is generated. In this type of signatures, the signer signs the message without ever seeing its contents: the user holds the message and interacts with the signer to obtain a signature. First, the user hides the message before sending it the the signer. Thus the message, and maybe suprisingly, the signature are not revealed to the signer. This process is akin to signing a sealed envelope made of carbon paper, where the signature is transferred onto the hidden message inside. Consequently, the signer remains unaware of the signature they have produced.

The primary use case for blind signatures was for a bank to issue a digital banknote without viewing its serial number, the main element that can be distinghuised between two bank notes of the same amount. Hence, the bank only knows the amount and the identity of the person withdrawing it and cannot trace the bank note through any means. This allows to remain anonymous in front of the bank, the merchant and any other kind of malicious entity observing the payment while paying. Hence, based on blind signatures one could obtain the same functionalities as today's electronic payments with better privacy, but this means of payment has not been perpetuated through the wild adoption of electronic payments.

Other types of authentication mechanisms, each with varying degrees of privacy or practical aspects, have also been introduced for other use cases. Some of the most studied types of privacy preserving types of signatures schemes are introduced below, when they are directly related to the main content of our studies. We go through a short retrospective.

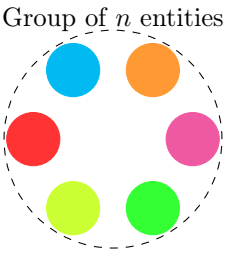


Figure 1.2: Seal of a Ring Signature [RST01].
(Linked to Chapter 3 on page 47)

Ring Signatures. Introduced by Rivest, Shamir and Tauman [RST01] in 1986, *ring signatures* are digital signatures whose allow an entity to sign in the name of an *ad hoc* group while concealing its identity from the verifier. We have schematically described the seal of this signature scheme in Figure 1.2, where several large dots represent several entities inside the ring, but this circle remains dashed because the ring in this type of signatures is *ad hoc*, *i.e.,* it is generated by the signer at the time of signing and has not required any form of acceptance from the other members included in the ring (apart from the generation of their public keys). This means that only one of the n signers (in Figure 1.2, $n = 6$) actually signed the message, the seal of the other group members was somehow mimiced by the actual signer. Ring signatures share the same expectation as regular signatures: they are difficult to forge without possessing the secret key. Additionally, any signature produced by a signer is indistinguishable from those generated by other members of the group, ensuring the anonymity of the signers. This primitive is related to the main theme of the Chapter 3 on page 47, in which we look at a primitive called *linkable ring signatures*. Linkable ring signatures balance the strong anonymity of ring signatures by linking signatures issued by the same signer. It is expected that two signatures from the same entity should always be linkable, while signatures from different entities must remain unlinked. These properties of linkable ring signature are complemented by those of ring signatures. This is visually described in Figure 1.3, each signer leaves its fingerprint while signing and they are the same while signing two different messages but differs when two entity signed. Here, we only used the color to distinguished the fingerprints, their is no link with the potential signer behind the signature. This is achieved for practical reasons, for example when we want to determine whether requests come from the same entity without needing its identity.

Figure 1.3: Seal of a Linkable Ring Signature and their Linkability [RST01].
(Linked to Chapter 3 on page 47)

Group Signatures. Introduced by Chaum and Heyst [CH91] in 1991, *group signatures* are another type of privacy preserving signatures which allow any member of a group to sign on behalf of one of the entities inside the group. The privacy of the signer is preserved and the signature can only be linked to the group and not the entity producing it. In this case, the group is managed by a central entity, commonly called *authority*, which owns the power to manage the group's registration and can de-anonymise any signature. The seal left by this type of signature is described schematically in Figure 1.4, assuming that the whole group and the authority are public information provided to the verifier. From the point of view of the verifier shown in Figure 1.4, it is impossible to tell which of the n entities in the group generated the signature. In the Figure, the symbole "A" denotes the authority and the circle is closed as all entities within the group must have been accepted by the authority before signing any message.

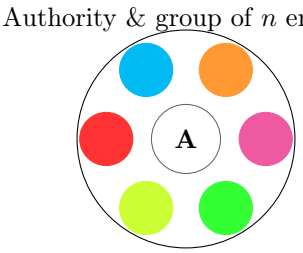


Figure 1.4: Seal of a Group Signature [CH91].
(Linked to Chapter 4 on page 77)

Proxy Signatures. Introduced by Mambo, Usuda, and Okamoto [MUO96] in 1996, *proxy signatures* enable one party, the *signer* also sometimes called the *original signer*, to delegate its signature rights to another party, named *proxy*, which can then sign documents on their behalf, facilitating secure delegation in various scenarios. It works in exactly the same way as a power of attorney for an election. In Figure 1.5, we can see the seal of the red signer giving delegation to the orange signer who put its own seal, the red part works like a certificate, like an evidence for the given delegation. The original concept was later extended in various ways. Our interest mainly goes to the primitive introduced by Fuchsbauer and Pointcheval [FP08] called *anonymous proxy signature*. These signatures provide anonymity for the proxy signer while maintaining the integrity and authenticity of the signed message. Hence, only the name of the original signer is disclosed.

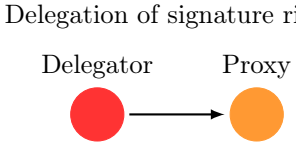


Figure 1.5: Seal of a Proxy Signatures.
(Linked to Chapter 4 on page 77)

We can see that many signature schemes with anonymity and various practical aspects have been developed. Here, we have ignored many other types of existing plans.

The ongoing research try to make them more efficient, secure, and improve their usability in a broad range of scenarios. Notably, there has been a push to develop a unified formalism encompassing most anonymous signature schemes [BDK24], amongst then group and ring signatures. However, as is often the case in cryptography, it is unrealistic to expect that these mechanisms will seamlessly fit into a single formalism. This highlights the need of further research to refine and develop the knowledge on these primitives and of their formalisation to, maybe, end-up with a unified definition in the future.

Above, we have discussed anonymity from a cryptographic perspective. However, it is useful to step back and examine it from a broader viewpoint. Digital and communication systems have been regulated ever since they first appeared, and numerous standards have been introduced. Systems originally designed from scratch now follow widely adopted frameworks. Introducing new designs requires adherence to legal aspects and existing standards and must offer improvements over existing systems to gain acceptance and later be deployed on our devices. In itself, designing a new protocol with improved functionality is a challenge, but when it comes to standard adoption and deployment, it's almost impossible. For example, a second worldwide payment system standard, aimed at significantly improving upon current protocols in terms of efficiency and mitigating numerous attacks, has been introduced in 2013. As of now, 11 years later, it is still not deployed (then, neither used), this shows the extensive time and willingness needed to alter established large-scale systems.

As this new norm awaits deployment, various proposals [HMY22, BHMY23] have been made to further enhance it, anticipating changes to what was originally proposed as the future standard.

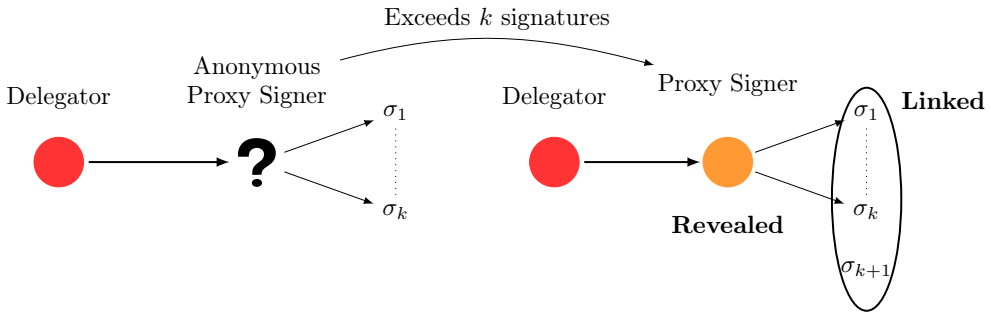
Anonymity was overlooked in the currently used payment service standard and in its second version: all identity related information are fully broadcast to all entities taking part in the payment. The authors of [HMY22, BHMY23] tried to address unlinkability against an eavesdropper's listening at the point of sale, this is already a step toward more privacy and security. However, it is not necessary for all the information to be known by everyone when a payment is made, as demonstrated by cash based transactions. The oversight of anonymity for customers against entities involved in payments highlights the need for further research and improvements in customer privacy. Especially as privacy concerns intensify and legal rights to privacy become more formalised, there is a critical need to ensure that future systems effectively safeguard user anonymity while also addressing fraud detection requirements.

1.3 Contributions of this Thesis

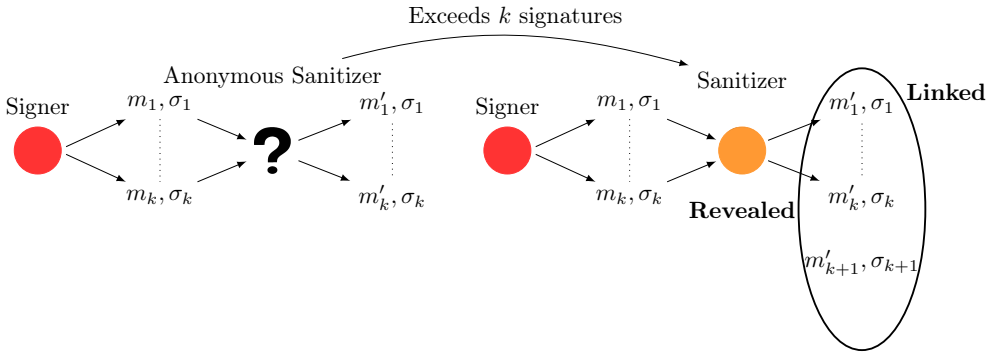
This thesis studies cryptography in the context of digital signature, proof technics in the computational model and anonymity in authentication. While setting the formal definitions of anonymity for signature schemes, it has been necessary to accommodate the degree of anonymity provided by each in order to meet the requirements of practical systems. In particular, we must acknowledge that full anonymity on the Internet has always been more of a concept than a reality. In most cases, full anonymity is too strong

for practical applications. It is in this context that we bring the three contributions of this thesis: (1) attempting to correct a shortcoming in the definition of anonymity of linkable ring signatures in Chapter 3. Almost all the previous definitions only guarantee anonymity for the first signature issued by a signer and do not take into account the anonymity of any further signatures produced, despite the fact that practical applications require this [LWW04]. Secondly, (2) we propose a new type of identity-preserving signature scheme in Chapter 4: *k-times fully anonymous proxy signatures*. In this type of signature, the proxy signer is anonymous, *i.e.,* does not reveal its identity unless it issues more than k signatures. If that limit is exceeded, its identity is *revealed* and can then be *linked* to all the signatures it has issued. We have visually represented these signatures in Figure 1.6a. In this figure, we see a delegation established by the red seal to an unknown entity allowed up to k signatures. If this entity exceeds the k authorised signatures by producing a $k+1^{\text{th}}$ signature, then its public key, associated to the orange color, is *revealed* and all the signatures produced are *linked*.

From our *k-times fully anonymous proxy signatures*, we were able to derive a *k-times fully anonymous sanitizable signature*. Sanitizable signatures acts relatively similarly to proxy signatures, through in addition that messages may be partially modified by the delegatee. We have visually represented these type signatures in Figure 1.6b and also introduce the first k-times fully anonymous sanitizable signatures scheme in Chapter 4.



(a) k-times Anonymous Full Traceable Proxy Signature (Section 4.3)



(b) k-times Anonymous Full Traceable Sanitizable Signature (Section 4.4)

Figure 1.6: k-times Anonymous Full Traceable Proxy and Sanitizable Signature.

Our third contribution (3) highlights a privacy-preserving architecture for card based payments compatible with both the current and future standards of payments. The study examines the global EMV (Europay Mastercard Visa) payment system, delv-

ing into more concrete system. It navigates the complexities of European and United Kingdom regulations to develop *privacy-enhancing*, *EMV-compatible*, *law-abiding*, and *usable* contactless payment protocols: `PrivBank` and `PrivProxy`. This exploration seeks to tackle some challenges toward implementing anonymity in practical applications. It also highlights the limitations and obstacles posed by various constraints and efforts lobbying against privacy.

Valorisation of this Thesis. The work carried out as part of this thesis has led to the production of several publications. Three articles, refered to below cover the work presented in this manuscript.

The various concepts presented in Chapter 2 of this thesis, the *technical background* provides the necessary preliminaries to support the reader in the subsequent chapters. It introduces notations, mathematical background, security notions, cryptographic assumptions, and primitives. These notions are then used to propose either new primitives or an architecture for improving the anonymity of a pre-existing protocol. We present the structure of the three main chapters of this manuscript below. They can be considered independently, but they are all linked by the themes they address and variations of what is generally refered to as anonymity.

[BOA24a] On the Anonymity of Linkable Ring Signatures, Xavier Bultel and Charles Olivier-Anclin. *Published in the proceedings of* CANS 2024.
Content of this work appears in Chapter 3.

[BOA24b] Taming Delegations in Anonymous Signatures: k-Times Anonymity for Proxy and Sanitizable Signature, Xavier Bultel and Charles Olivier-Anclin. *Published in the proceedings of* CANS 2024.
Content of this work appears in Chapter 4.

[BCC$^+$24] EMV-Compliant and Usable Anonymity for Contactless Payments, Ioana Boureanu, Liqun Chen, Tom Chothia, Anna Clee, Andreas Kokkinis, Pascal Lafourcade, Chris Newton, and Charles Olivier-Anclin. *Published in the proceedings of* USENIX Security 2025.
Content of this work appears in Chapter 5.

A conclusion summarizing the contributions of the work presented in this thesis is provided at the end, in Chapter 6.

1.4 Other Published Work

Other research works were carried out during during this PhD but is left uncovered by this manuscript, we list such contributions below with a brief abstract for each.

[LMOAR24] Secure Keyless Multi-Party Storage Scheme,
Pascal Lafourcade, Lola-Baie Mallordy, Charles Olivier-Anclin and Léo Robert, *Published in the proceedings of* ESORICS 2024

Abstract. Using threshold secret sharing, we propose a solution tailored for *forgetful* clients (*i.e.,* not required to keep any cryptographic secret) while accommodating the dynamic nature of multi-cloud deployments. Furthermore, we delegate the computation and distribution of shares to an intermediate server (proxy), effectively minimizing the client workload. We propose two variants of a keyless, space-efficient multi-cloud storage scheme named KAPRE and KAME. Our solution KAPRE requires less communications and computations, while KAME preserves data confidentiality against a colluding proxy. Our protocols offer robust guarantees for data integrity, and we demonstrate the proxy's ability to identify and attribute blame to servers responsible for sending corrupted shares during data reconstruction. We establish a comprehensive security model and provide proofs of the security properties of our protocols. To complement this theoretical analysis, we present a proof-of-concept to illustrate the practical implementation of our proposed scheme.

[LMMOA24] Transferable, Auditable and Anonymous Ticketing Protocol,
Pascal Lafourcade, Dhekra Mahmoud, Gael Marcadet and Charles Olivier-Anclin,
Published in the proceedings of ASIACCS 2024

Abstract. Digital ticketing systems typically offer ticket purchase, refund, validation, and, optionally, anonymity of users. However, it would be interesting for users to transfer their tickets, as is currently done with physical tickets. We propose Applause, a ticketing system allowing the purchase, refund, validation, and transfer of tickets based on trusted authority, while guaranteeing the anonymity of users, as long as the used payment method provides anonymity. To study its security, we formalise the security of the transferable E-Ticket scheme in the game-based paradigm. We prove the security of Applause computationally in the standard model and symbolically using the protocol verifier ProVerif. Applause relies on standard cryptographic primitives, rendering our construction efficient and scalable, as shown by a proof-of-concept. In order to obtain Spotlight, an auditable version, proved to be secure, users will remain anonymous except for a trusted third party, which will be able to disclose their identity in the event of a disaster.

[AHKLOA23] Generic Privacy Preserving Private Permissioned Blockchains,
Frédéric A. Hayek, Mirko Koscina, Pascal Lafourcade and Charles Olivier-Anclin,
Published in the proceedings of SAC 2023

Abstract. Private permissioned blockchains are becoming gradually more sought-after. Such systems are reachable by authorised users, and tend to be completely transparent to whomever interacts with the blockchain. In this paper, we mitigate the latter. Authorised users can now stay unlinked to the transaction they propose in the blockchain while being authenticated before being allowed to interact. As a first contribution, we developed a consensus algorithm for private permissioned blockchains based on Hyperledger Fabric and the Practical Byzantine Fault Tolerance consensus. Building on this blockchain, five additional variations achieving various client-wise privacy preserving levels are proposed. These different protocols allow for different use cases and levels of privacy control and sometimes its revocation by an authority. All our

protocols guarantee the unlinkability of transactions to their issuers achieving anonymity or pseudonymity. Miners can also inherit some of the above privacy preserving setting. Naturally, we maintain liveness and safety of the system and its data.

[BBC+23] Practical Construction for Secure Trick-Taking Games Even With Cards Set Aside, Rohann Bella, Xavier Bultel, Céline Chevalier, Pascal Lafourcade and Charles Olivier-Anclin,
Published in the proceedings of FC 2023.

Abstract. Trick-taking games are traditional card games played all over the world. There are many such games, and most of them can be played online through dedicated applications, either for fun or for betting money. However, these games have an intrinsic drawback: each player plays its cards according to several secret constraints (unknown to the other players), and if a player does not respect these constraints, the other players will not realise it until much later in the game.

In 2019, X. Bultel and P. Lafourcade proposed a cryptographic protocol for Spades in the random oracle model allowing peer-to-peer trick-taking games to be played securely without the possibility of cheating, even by playing a card that does not respect the secret constraints. However, to simulate card shuffling, this protocol requires a custom proof of shuffle with quadratic complexity in the number of cards, which makes the protocol inefficient in practice. In this paper, we improve their work in several ways. First, we extend their model to cover a broader range of games, such as those implying a set of cards set aside during the deal (for instance Triomphe or French Tarot). Then, we propose a new efficient construction for Spades in the standard model (without random oracles), where cards are represented by partially homomorphic ciphertexts. It can be instantiated by any standard generic proof of shuffle, which significantly improves the efficiency. We demonstrate the feasibility of our approach by giving an implementation of our protocol, and we compare the performances of the new shuffle protocol with the previous one. Finally, we give a similar protocol for French Tarot, with comparable efficiency.

[KLM+22] A Survey on Identity-based Blind Signature, Mirko Koscina, Pascal Lafourcade, Gael Marcadet, Charles Olivier-Anclin, Léo Robert
Published in the proceedings of FPS 2022.

Abstract. Blind signatures are well-studied building blocks of cryptography, originally designed to enable anonymity in electronic voting and digital banking. Identity-based signature were introduced by Shamir in 1984 and gave an alternative to prominent Public Key Infrastructure. An identity-based blind signature (IDBS) allows any user to interact directly with the signer without any prior interaction with a trusted authority. The first IDBS has been proposed in 2002 and several schemes were proposed since then. Seeking for a full comparison of these primitives, we propose a survey on IDBS and list all such primitives that seems to maintain some security. We also classify their security assumptions based on the existing security expectation that have not been formalised yet in the literature. Moreover, we empirically evaluate the complexity of all the operations used in those schemes with modern cryptographic libraries. This allows us to perform a

realistic evaluation of their practical complexities. Hence, we can compare all schemes in terms of complexity and signature size.

[BLOR21] Generic Construction for Identity-based Proxy Blind Signature, Xavier Bultel, Pascal Lafourcade, Charles Olivier-Anclin, Léo Robert,
Published in the proceedings of FPS 2021.

Abstract. Generic constructions of blind signature schemes have been studied since its appearance. Several constructions were made leading to generic blind signatures and achieving other properties such as identity-based blind signature and partially blind signature. We propose a generic construction for identity-based Proxy Blind Signature (IDPBS). This combination of properties has several applications in the real world, in particularly in e-voting or e-cash systems and it has never been achieved before with a generic construction. Our construction only requires two classical signatures schemes: a blind EUF-CMA blind signature and a SUF-CMA unique signature. The security of our generic identity-based proxy blind signature is proven under these assumptions.

Chapter 2

Technical Background

Contents

Chapter Summary

This chapter provides an overview of the essential concepts and notations required for our study. We also discuss the main cryptographic primitives used in our work such as digital signatures, zero-knowledge proofs and hash functions, which are essential for guaranteeing the confidentiality, integrity and authenticity of data.

To guarantee data security, cryptography aims to ensure, amongst other properties, confidentiality, authenticity and integrity. To determine whether an encryption or signature scheme meets these properties, researchers use problems that are considered computationaly difficult and reduce the security of their schemes to the difficulty of these problems. Meaning that if an attacker were to break such a protocol, it would imply that it had solved one of these difficult problems, many of which have been studied for years and for which the complexity of even the most efficient algorithms is still not enough to solve it efficiently. These assumptions are commonly referred to as *security assumptions*. Our cryptographic constructions follow the same principle and are based on classical problems presented below. To better formalise the overall concept, we will present the formalism underlying cryptographic constructions, these assumptions and define some well-studied constructions.

2.1 Notations

We begin by presenting the most common notations used in this manuscript. Some notations may also appear later.

- Let $\mathbb{N}$ and $\mathbb{Z}$ respectively denote the sets of positive and relative integers.
- Let 1^λ be the unary representation of $\lambda \in \mathbb{N}$.
- Let $\mathbb{Z}_p$ for $p \in \mathbb{Z}$ denote the ring of integers modulo p.
- Let $[\![n]\!]$ denote the set of integers $\{1, \dots, n\}$.
- Let $|S|$ denote the cardinality of a set S.
- Let $s \xleftarrow{\$} S$ denote the uniform sampling of s over the set S.
- Let $X \xrightarrow{p} (x_i)_{i \in [\![n]\!]}$ denote the parsing of a tuple or a set of n elements.
- Let $y \leftarrow \mathsf{Alg}(x)$ denote the execution of an algorithm Alg outputting y on input x.
- Let $\mathcal{A}^{\mathcal{O}}$ denote an algorithm $\mathcal{A}$ that can call a subroutine/oracle $\mathcal{O}$.
- Let $[\mathsf{Alg}]$ denote the set of all possible outputs of the algorithm Alg.
- Let $\sqcup$ denote the union $\cup$ when applied to multi-sets, thereby preserving the repetition of elements.
- Let $\eta[i]$ denote the i^{th} bit of the binary representation of the integer $\eta \in \mathbb{N}$. This representation is fixed using the canonical bijection between $\mathbb{N}$ and $\{0,1\}^*$, given by the equality $\eta = \sum_{i=0}^{n-1} 2^i \cdot \eta[i]$.

2.2 Computational Background

In what follows, we introduce the fundamental concepts of computational complexity necessary to formalise the security guarantees of cryptographic primitives in the computational model. This discussion is grounded in polynomial-time reductions and the fundamental security assumptions that support these guarantees.

2.2.1 Algorithms Properties

Definition 1: Negligible Function

A function $\epsilon \colon \mathbb{N} \to \mathbb{R}$ is *negligible* if for every positive integer $c \in \mathbb{N}$, there exists an integer $N_c \in \mathbb{N}$ such that for all $x > N_c$, $|\epsilon(x)| < 1/x^c$.

Definition 2: Polynomial Algorithm

An algorithm Alg with input of size $\lambda \in \mathbb{N}$ is said to be *polynomial-time*, if for every input of said size, there exists a polynomial $P \in \mathbb{N}[X]$ such that the number of operations executed by Alg is bounded by $P(\lambda)$.

Note that the word *operation* has not been defined. The rigorous formalisation comes from the formalisation of mathematical models of computation such as *Turing machine theory* in complexity theory. In this work, we do not introduce it from scratch and refer to [Gol01, GMR19] for full definitions. What we call operations can be viewed as gate operations at processor level or as additions or multiplications of integers when there is a mathematical structure. Indeed, these operations have a polynomial complexity in the number n of bits required to encode the integer involved. Additions require $O(n)$ operation while multiplications require $O(n^2)$ with the schoolbook multiplication down to $O(n \log(n))$ for the best known algorithm, the Harvey-Hoeven

algorithm [HVDH21]. Therefore, performing a polynomial number of additions or multiplications is a polynomial-time algorithm.

Algorithms may include a degree of uncertainty in their results, even when they are executed multiple times on the basis of the same inputs. This is what we call a probabilistic algorithm.

Definition 3: Probabilistic Algorithm

Let Alg be a algorithm taking inputs in X and producing elements in Y. It is said to be *probabilistic* if there exists $x \in X$ such that $|[\mathsf{Alg}(x)]| > 1$. Equivalently formulated, given two executions $\mathsf{Alg}(x) \to y$ and $\mathsf{Alg}(x) \to y'$, the results y and y' have a non-zero probability of being different.

The *security parameter* $\lambda \in \mathbb{N}$ is a variable that measures the input size of a computational problem, *i.e.,* the number of bits required to represent the problem. An adversary against a given problem is assumed to be a polynomial-time probabilistic algorithm running in a time that is polynomial in λ. In this case, security must be guaranteed except with a negligible probability in the security parameter.

2.2.2 Provable Security

Technical Summary

Based on the notion of a polynomial-time algorithm, we formulate cryptographic properties in the so-called *computational model*. A *security reduction* refers to a mathematical proof that shows that solving one problem, usually in cryptography the task of compromising the security a scheme, is at least as difficult as solving another, typically well-established problem believed to be computationally unfeasible. The *quality* of a security reduction quantifies how effectively this reduction translates the difficulty of breaking the cryptographic scheme under consideration to the known hard problem. Various factors influence the assessment of the quality of a security reduction, which aids in evaluating the practical security of a cryptographic scheme. We discuss these elements below.

Let $\mathcal{A}$ be any probabilistic (most of the time also polynomial-time and then PPT) algorithm, called the *adversary*, with prescribed inputs and outputs (the latter being specified according to context). We consider another PPT algorithm Exp, called *experiment* featering an entity called the *challenger* $\mathcal{C}$ and executing $\mathcal{A}$ as a subroutine, or alternatively interacting with $\mathcal{A}$. The cryptographic properties in the computational model are defined on the basis of the experiment Exp, and are said to be realised by a scheme S, or not, by observing the probability of occurrence of its outputs. In an experiment, we generally denote the elements returned by the adversary $\mathcal{A}$ with a star in superscript *e.g.,* "σ^*". The general notation, in this manuscript, for an experiment associated to a property prop is the following:

$$\mathsf{Exp}^{\mathsf{prop}}_{\mathcal{A},\mathsf{S}}(1^\lambda) \to \mathsf{out}.$$

In general, these experiments fall within two categorises:

Computation/Extraction Problem. In this type of problem, we observe if there exists a PPT algorithm $\mathcal{A}$ capable of outputting an element of a prescribed form for a given scheme S. $\mathcal{C}$ generates an execution environment for $\mathcal{A}$ and verifies if the value outputted by $\mathcal{A}$ matches the expected properties. Here the output of $\mathsf{Exp}^{\mathsf{prop}}_{\mathcal{A},\mathsf{S}}(1^\lambda)$ is either 0 (failure) or 1 (success). For the property prop to be achieved, we require that for a negligible function $\epsilon(1^\lambda)$ in the security parameter λ:

$$\mathsf{Adv}^{\mathsf{prop}}_{\mathcal{A},\mathsf{S}}(1^\lambda) = \Pr[\mathsf{Exp}^{\mathsf{prop}}_{\mathcal{A},\mathsf{S}}(1^\lambda) = 1] = \epsilon(1^\lambda).$$

$\mathsf{Adv}^{\mathsf{prop}}_{\mathcal{A},\mathsf{S}}(1^\lambda)$ is called the *advantage* of the adversary $\mathcal{A}$ against the property prop of the scheme S for the security parameter λ.

Decisional Problem. In this type of problem, we observe if there exists a PPT algorithm $\mathcal{A}$ capable of distinguishing between two events. Once more, $\mathcal{C}$ generates an execution environment for $\mathcal{A}$ and based on a random bit b sampled uniformly at random in $\{0, 1\}$, it executes either the scenario 0 or 1. It is expected that $\mathcal{A}$ determines which of the scenarios is executed. The experiment $\mathsf{Exp}^{\mathsf{prop}}_{\mathcal{A},\mathsf{S}}(1^\lambda)$ still outputs 0 or 1. For the property prop to hold, we require that:

$$\mathsf{Adv}^{\mathsf{prop}}_{\mathcal{A},\mathsf{S}}(1^\lambda) = |\Pr[\mathsf{Exp}^{\mathsf{prop}}_{\mathcal{A},\mathsf{S}}(1^\lambda) = 1] - 1/2]| = \epsilon(1^\lambda).$$

Showing that a given property prop is achieved by a scheme S and for all potential algorithms $\mathcal{A}$ is achieved by a *security reduction*. It consists of finding a polynomial reduction from breaking a given property of a scheme to solving a (presumably) difficult problem. This applies if we can provide a polynomial-time algorithm that breaks the hard problem if there is a PPT algorithm that breaks the property of the scheme. Combining a polynomial-time reduction with a polynomial-time adversary against the system property would then lead to a polynomial method for solving the supposedly difficult problem. Hence, we would obtain a contradiction if the problem is indeed hard to solve. In practice, it is not known whether such a contradiction actually exists, as it would require the existence of a problem in the NP complexity class that cannot be solved in polynomial-time. The existence of such a problem has not been demonstrated.

One of the main methods used to find reductions for complex systems is employing *sequence of games* [Sho04]. This consists of introducing, one by one, a sequence of small differences into the challenge given to an adversary. Consider an *experiment* $\mathsf{Exp}^{\mathsf{prop}}_{\mathcal{A},\mathsf{S}}(1^\lambda)$ for a given property prop, an adversary $\mathcal{A}$ and a scheme S. The small changes provide a sequence $\mathsf{Exp}^{\mathsf{G}_0}_{\mathcal{A},\mathsf{S}}(1^\lambda)(= \mathsf{Exp}^{\mathsf{prop}}_{\mathcal{A},\mathsf{S}}(1^\lambda))$, …, $\mathsf{Exp}^{\mathsf{G}_n}_{\mathcal{A},\mathsf{S}}(1^\lambda)$ where $\mathsf{Exp}^{\mathsf{G}_i}_{\mathcal{A},\mathsf{S}}(1^\lambda)$ and $\mathsf{Exp}^{\mathsf{G}_{i+1}}_{\mathcal{A},\mathsf{S}}(1^\lambda)$ are always closely related. To simplify the notation, we simply denote the experiment $\mathsf{Exp}^{\mathsf{G}_i}_{\mathcal{A},\mathsf{S}}(1^\lambda)$ by $\mathsf{G}_i(\mathcal{A})$ or even G_i when the adversary is clear in the context. These sequence are obtained by each time introducing small modifications in an experiment G_i to produce the experiment G_{i+1}. This sequence of successive small changes is often referred to as a *game hop* [Sho04]. These games which follow one another can be related in various ways.

Game Hop Based on Indistinguishability. If detected by an adversary, these changes imply the existence of an adversary capable of distinguishing between two (assumed) indistinguishable distributions (either statistically or computationally). For example, consider two successive experiments G_i and G_{i+1} in a sequence and a well-established decisional problem. When we seek to prove that,

$$|\Pr[\mathsf{G}_i(\mathcal{A}) = 1] - \Pr[\mathsf{G}_{i+1}(\mathcal{A}) = 1]| \leq \Pr[\mathcal{D} \text{ solves the difficult decisional problem}].$$

In this order, we consider an adversary $\mathcal{A}$ with non-negligible probability of distinguishing between G_i and G_{i+1}, *i.e.,* on the left-hand side of the inequality, and through a polynomial reduction we can construct a *distinguishability algorithm* $\mathcal{D}$ using $\mathcal{A}$ as a subroutine to find the solution to the hard decisional problem. This creates a contradiction with the main hypothesis of the hardness of the decisional problem. If we relied on a perfectly indistinguishable problem, this results to $\Pr[\mathsf{G}_i(\mathcal{A}) = 1] = \Pr[\mathsf{G}_{i+1}(\mathcal{A}) = 1]$.

Game Hop Based on Failure Events. Transitions between games can also be achieved by introducing *failure events*. Consider a failure event E. Given an experiment G_i in a game hop, we instantiate the experiment G_{i+1} so that it is similar to G_i but at some point during its execution, if the failure event E is encountered, the experiment fails and returns a failure otherwise it proceeds exactly like G_i. If this event E is never encountered, the experiments G_i and G_{i+1} remain identical, leading to an equal probability of success:

$$\Pr[\mathsf{G}_i \cap \neg E] = \Pr[\mathsf{G}_{i+1} \cap \neg E].$$

From this equality of probabilities, it follows that:

$$|\Pr[\mathsf{G}_i] - \Pr[\mathsf{G}_{i+1}]| \leq \Pr[E].$$

This formula states that if we considered an event E occurring with negligible probability, then the difference of success between G_i and G_{i+1}, as described above, is also negligible.

Game Hop Based on Polynomial Probability. We also employ a proof technique based on guessed choice events, made by the challenger, with probability P which is polynomial as a function of the security parameter λ. For instance, given two indistinguishable users, the challenger selects one at random and expects an attack to be conducted against this chosen individual. In this case, $P(\lambda) = 1/2$. Now, if we consider a polynomial P in the security parameter λ, and assume that two games are related as $\Pr[\mathsf{G}_i] = P(\lambda) \cdot \Pr[\mathsf{G}_{i+1}]$, with the event G_{i+1} occurring with negligible probability, that is $\Pr[\mathsf{G}_{i+1}] = \epsilon(1^\lambda)$, it follows that $\Pr[\mathsf{G}_i] = \frac{1}{P(\lambda)} \cdot \epsilon(1^\lambda)$, which is also a negligible quantity.

Bridging steps. These steps are a way of restating how certain elements are computed without changing their distribution, *i.e.,* the change does not affect any of the distribution of any of the elements, then $\mathsf{Diff}^{\mathsf{prop}}_{\mathsf{G}_i\text{-}\mathsf{G}_{i+1}}(1^\lambda) = 0$. The purpose of these steps is double: to prepare future transitions usually in order to be able to execute a reduction

to an Indecisional problem or a failure event and to make the proof easier to follow and be verified.

For all $i \in [\![n-1]\!]$, we denote by:

$$\mathsf{Diff}^{\mathsf{prop}}_{\mathsf{G}_i\text{-}\mathsf{G}_{i+1}}(1^\lambda) = |\Pr[\mathsf{G}_i(\mathcal{A}) = 1] - \Pr[\mathsf{G}_{i+1}(\mathcal{A}) = 1]|\,,$$

the difference in $\mathcal{A}$'s success probability in games G_i and G_{i+1}. When it is shown that for all $i \in [\![n-1]\!]$, $\mathsf{Diff}^{\mathsf{prop}}_{\mathsf{G}_i\text{-}\mathsf{G}_{i+1}}(1^\lambda) \leq \epsilon_i(1^\lambda)$, and that $\mathcal{A}$ cannot win against G_n with probability gretter than $\epsilon_n(1^\lambda)$, then we can conclude that $\mathsf{Adv}^{\mathsf{prop}}_{\mathcal{A},\mathsf{S}}(1^\lambda) = \sum_{i\in[\![n]\!]} \epsilon_i(1^\lambda)$ is negligible, as the finite sum of negligible functions is also negligible.

Another proof technique used in this manuscript is the *hybrid argument* [FM21]. A hybrid argument is a fundamental proof technique used to show the indistinguishability of two probability distributions, here the distributions of M_0 and M_1. It is based on a game hop $H_0, \ldots, H_t$, for $t \in \mathbb{N}$ (here t is a constant value, but it can be up to polynomial in the security parameter of the scheme [FM21]), such that the difference between any H_i and H_{i+1} is always the same negligible change (up to indexation) and the probability of winning against the last game is negligible.

Assessing the quality and relevance of a security reduction provided by a game hop comes down to several key factors; which we list below.

Adversarial Model. A reduction is meaningful only when it relies on *realistic* assumptions on the adversary trying to break the system. This means that it must accurately take into account the capabilities of potential attackers. A robust adversarial model takes into account all the possible methods that an adversary could use. One of the most widely used models in black box cryptography is the *standard model*, which describes the adversary as any probabilistic polynomial-time algorithm interacting with a challenger. Other adversarial models with more restricted adversaries are discussed in more detail in Section 2.4.2.

Tightness of the Reduction. A reduction is considered *tight* if the security guarantee of the cryptographic scheme closely matches (up to a constant factor) the assumed difficulty of the underlying problem. For instance, if breaking a cryptosystem with a probability p translates into solving an underlying problem with a probability $c \cdot p$ for a constant c expected to be as small as possiblen then this is a tight reduction since:

$$\Pr[\mathcal{A} \text{ breaks the scheme}] = c \cdot \Pr[\mathcal{B} \text{ solves the hard problem}].$$

The chances of solving the difficult problem in the event of an adversary breaking the property of the scheme are directly proportional for any given security parameter. However this is not always the case. Some reductions show that breaking the cryptosystem with probability p results in solving the underlying problem with probability p/q, where q is a polynomial in the security parameter. This is still a polynomial reduction and p/q is proportional to p but the probability of finding a solution to the hard problem is significantly smaller than p. When q has a high degree polynomial, the reduction is considered *loose*. It should be emphasised that there is no clear characterisation in the

literature defining when a reduction becomes loose, though it remains a polynomial reduction. Loose reductions typically provide less confidence in the security of the scheme compared to tight reductions.

Efficiency of the Reduction. The reduction should be possible to carry out. For example, if a reduction requires large lookup tables to be stored, then a large amount of memory is needed to run the algorithm prescribed by the reduction in order to solve the hard problem. Running the algorithm may even be unpractical despite the potential polynomial reduction that has been achieved if the memory required is too big.

Assumption Strength. The quality of a proof also depends on the assumptions it makes. Assumptions grounded in widely accepted hard problems (*e.g.,* the hardness of factoring large integers or the discrete logarithm problem) lend more credibility and robustness to the security guarantee provided by the reduction than thoses grounded in less studied problems.

The emergence of quantum computers has opened up new perspectives and the long-standing problem could be abandoned in favour of new problems that would withstand the algorithms executed by this new type of computer. We do not discuss these problem whithin this work as it is out of the scope of this work. In general formalising new anonymity properties transcend cryptographic security assumptions even so the instantition of the schemes might not.

2.3 Mathematical Background

We will now examine the mathematical context and requirements of the following chapters. We begin with the notion of groups.

Definition 4: Group

A group $(\mathbb{G}, \cdot)$ is a pair where $\mathbb{G}$ is a set of elements and $\cdot : \mathbb{G} \times \mathbb{G} \to \mathbb{G}$ a function verifying:

Associativity. For all a, b and $c \in \mathbb{G}$, $(a \cdot b) \cdot c = a \cdot (b \cdot c)$.

Identity Element. There exists an element $1_{\mathbb{G}} \in \mathbb{G}$ called the *identity element*, such that for all $a \in G$, $1_{\mathbb{G}} \cdot a = a = a \cdot 1_{\mathbb{G}}$. This element will be abbreviated later as 1.

Inverse Element. For all $a \in \mathbb{G}$, there exists an unique element $a^{-1} \in \mathbb{G}$ such that $a \cdot a^{-1} = 1_{\mathbb{G}} = a^{-1} \cdot a$.

We generally refer to a group only by means of its set of elements $\mathbb{G}$. Let $g \in \mathbb{G}$ and $k \in \mathbb{N}$, denote $g^k = g \cdot \ldots \cdot g$ as the repetition of the operation $\cdot$ to k elements equal to g. For a vector $m = (m_1, \ldots, m_n) \in \mathbb{G}^n$ and an integer μ, the notation m^μ indicates the vector $(m_1^\mu, \ldots, m_n^\mu)$ where all the operations have been applied to all elements independently.

Definition 5: Generator of a Group

Let $(\mathbb{G}, \cdot)$ be a finite group ($\mathbb{G}$ is a finite set). We say that g is a generator of $(\mathbb{G}, \cdot)$ if $\mathbb{G} = \{g^k | k \in \mathbb{Z}\}$. If such an element $g \in \mathbb{G}$ exists, the group $\mathbb{G}$ is called *cyclic*.

Definition 6: Equivalent Classes

An equivalence relation $\mathcal{R}$ on a set X is a binary relation satisfying the three following properties:

Reflexivity. $a \mathcal{R} a$ for all $a \in X$.

Symmetry. $a \mathcal{R} b$ implies $b \mathcal{R} a$ for all $a, b \in X$.

Transitivity. if $a \mathcal{R} b$ and $b \mathcal{R} c$ then $a \mathcal{R} c$ for all $a, b, c \in X$

The equivalence class of an element $a \in X$ is denoted as $[a]_{\mathcal{R}} = \{x \in X : a \mathcal{R} x\}$. We may omit the relation $\mathcal{R}$ when it is clear from the context. In this manuscrit, only one is considered : $\mathcal{R} = \{(m, m') \in \mathbb{G}^l \times \mathbb{G}^l \mid \exists \mu \in \mathbb{Z}_p, m' = m^\mu\}$.

Definition 7: Bilinear Pairing

Let $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_t$ be three groups of prime order p. A *bilinear pairing* map $e\colon \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_t$ is a polynomial-time computable mapping, satisfying the following properties:

Bilinearity. For all $g_1 \in \mathbb{G}_1$, $g_2 \in \mathbb{G}_2$, for all $a, b \in \mathbb{Z}$, it holds that $e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}$.

Non-degeneracy. For all $g_1 \in \mathbb{G}_1$ and $g_2 \in \mathbb{G}_2$, it holds that $e(g_1, g_2) = 1 \implies g_1 = 1$ or $g_2 = 1$.

Bilinear pairings have been classified based on the relation between their base groups $\mathbb{G}_1$ and $\mathbb{G}_2$.

Type 1. $\mathbb{G}_1 = \mathbb{G}_2$, these pairing are said to be *symmetric*.

Type 2. $\mathbb{G}_1 \neq \mathbb{G}_2$ and there exists a polynomial-time homomorphism $\phi\colon \mathbb{G}_1 \to \mathbb{G}_2$.

Type 3. $\mathbb{G}_1 \neq \mathbb{G}_2$ and their is no known polynomial-time homomorphism between $\mathbb{G}_1$ and $\mathbb{G}_2$.

In this manuscript we only consider pairing of type 3 as it required for specific primitives used in this work such as the SPS signature scheme from [FHS19] described in Figure 2.1. Moreover it seems to be stronger against existing attacks.

2.4 Cryptographic Background

Cryptographic assumptions are presumably intractable problems that form the foundation of provably secure cryptography. We provide the assumptions necessary for our

work in Section 2.4.1. They form the basis of the security of the primitives introduced in Section 2.4.2, which are then used as our building blocks in Chapter 3 and 4.

2.4.1 Cryptographic Assumptions

One of the fundamental problems in cryptography is named after *Whitfield Diffie* and *Martin Hellman*. It was originally proposed in their seminal work [DH76].

Definition 8: Decision Diffie-Hellman (DDH) problem

Let $\mathbb{G}$ be a group generated by a random generator g. Let a and b be two random values in $\mathbb{Z}_{|\mathbb{G}|}$. Given $(g, g^a, g^b, g^z) \in \mathbb{G}^4$, the *Decisional Diffie-Hellman* (DDH) problem requires determining whether $z = a \cdot b$ or z is sampled uniformly at random.

The hardness of the DDH problem implies hardness of the *Discrete Logarithm* (DL) problem.

Definition 9: Discrete Logarithm (DL) problem

Let $\mathbb{G}$ be a group generated by a random generator g. Let a and b be two random values in $\mathbb{Z}_{|\mathbb{G}|}$. Given $(g, g^x) \in \mathbb{G}^2$, the *Discrete Logarithm* (DL) problem require to return x with non-negligible probability.

Now, consider the relation $\mathcal{R}$ over a group $\mathbb{G}$ of prime order p defined by $\mathcal{R} = \{(m, m') \in \mathbb{G}^l \times \mathbb{G}^l \mid \exists \mu \in \mathbb{Z}_p, m' = m^\mu\}$ implying an equivalence classes $[M]_\mathcal{R} \subset \mathbb{G}$ for an element $M \in \mathbb{G}^l$. We can formulate the *Class-hiding* problem as follows.

Definition 10: Class-hiding

Let $l > 1$ be an integer, and $\mathbb{G}$ a group. $\mathbb{G}$ is *class-hiding* if for all PPT adversaries $\mathcal{A}$, the following probability is negligible:

$$\Pr\begin{bmatrix} b \xleftarrow{\$} \{0,1\},\ M \xleftarrow{\$} (\mathbb{G}^*)^l,\ M^{(0)} \xleftarrow{\$} (\mathbb{G}^*)^l, \\ M^{(1)} \xleftarrow{\$} [M]_\mathcal{R},\ b^* \leftarrow \mathcal{A}(M, M^{(b)}) \end{bmatrix} : b = b^* \Bigg].$$

Lemma 1: Fuchsbauer *et al.* [FHS19]

Let $l > 1$ be an integer, and $\mathbb{G}$ be a group of prime order p. Then $\mathbb{G}$ is a class-hiding if and only if the DDH assumption holds in $\mathbb{G}$.

2.4.2 Cryptographic Models

To enable security proofs, models defining the adversary's capabilities have been studied and offer various guarantees and security levels. They range from the standard model, which imposes no restrictions on the adversary's algorithm capabilities excepting, in most, but not all cases, the limitation of its computational power: "*the adversary $\mathcal{A}$ is a probabilistic polynomial-time algorithm*", to more restricted models such as the *Random*

Oracle Model (ROM) [BR93] or the Generic Group Model (GGM) [Mau05]. We describe them below. We discuss the ROM as most construction introduced in the Chapter 3 and Chapter 4 rely on it and the GGM, because our schemes of Chapter 4 use a building block whose security has been proven in this model. This building block comes from [FHS19] and is described in Figure 2.1. Therefore, the security of our signatures of Chapter 4 depends on it.

Technical Summary

The ROM and GGM are heuristic models used to carry out cryptographic security arguments. Their use limits the confidence placed in the proof of security. In the ROM, cryptographic signature schemes or encryption schemes can be shown to be secure, whereas when instantiated with any real hash function, trivial attacks become possible [CGH04, GR04]. On the other hand, the GGM assumes a limitation on the adversary's capabilities and therefore does not cover the full range of actions that could be performed by an adversary. However, no non-artificial scheme proven within these models has ever been shown to be unsafe. The preferred model is therefore the standard model, in which no heuristic have been made about the adversary or other parts of the system, and the other models still provide confidence in the security of the schemes.

Hash Functions and the Random Oracle Model.

Definition 11: Hash Function

A cryptographic *hash function* $H\colon \{0,1\}^* \to \{0,1\}^\lambda$ is a deterministic polynomial-time algorithm mapping an arbitrary-length message to a bitstring of fixed size λ, called the *hash value* or *message digest*. Cryptographic hash functions are assumed to be *collision resistant* as defined below.

Collision Resistant [Dam87]: there is no known polynomial-time algorithm capable of finding $x \in \{0,1\}^*$ and $x' \in \{0,1\}^*$ such that $H(x) = H(x')$ and $x \neq x'$ with non negligible probability.

The *Random Oracle Model* (ROM) [BR93] is an heuristic often used in order to prove the security of cryptographic primitives based on hash functions. It provides an idealised hash function, which providies uniformly distributed random elements for each input queried to the function and returns consistent answers when queried twice with the same input. For this function to be instantiated it would require *mapping tables*, *i.e.,* arrays, to store the input queries with the corresponding output. The description of such a function is exponential in the size of the inputs and outputs. It follows that such a function may only be an idealised object, as we do not know if efficiently computable functions with this property do exist. In practice, real-world applications rely on cryptographic hash functions to instantiate random oracles.

Under this heuristic model, some artificial signature and encryption schemes can be proven secure, any instantiation thereforce using a *non-ideal* hash function would makes them vulnerable [CGH04, GR04]. This demonstrate that a proof in random oracle model does not guarantee the same security as the standard model. However, no proven non-

artificial scheme in the ROM has yet been broken, hence this model seems to provide strong evidence of security [KM15]. Hence, a security proof in the random oracle model can indicate the security of a cryptographic protocol, through it remains weaker in terms of guarantiees than a proof in the standard model.

Definition 12: Random Oracle Model

A cryptographic scheme is secure in the *random oracle model* (ROM) when its security proofs requires that any hash function $H\colon \{0,1\}^* \to \{0,1\}^\lambda$ be replaced by a black box function H (called the random oracle) defined as follows for a set Out initialised as the empty set (*i.e.,* $\mathsf{Out} \leftarrow \emptyset$):

Oracle $H(m)$	
1 :	**if** $\exists h \in \{0,1\}^\lambda, (m,h) \in \mathsf{Out}$,
2 :	**return** h
3 :	$h \xleftarrow{\$} \{0,1\}^\lambda$
4 :	$\mathsf{Out} \leftarrow \mathsf{Out} \cup \{m,h\}$
5 :	**return** h

In this manuscript, we prove specific security properties within the framework of the random oracle model.

Generic Group Model (GGM). The Generic Group Model (GGM) [Mau05] abstracts the capabilities of an adversary by restricting their computations to operations within a group, without allowing them to exploit the group underlying structure. The model assumes that group elements do not reveal any information about the group's underlying structure that could be used to break cryptographic problems. To this end, an oracle provides random encodings of group elements that remain consistent across queries, and algorithms accessing these encodings can only perform group operations or check for equality. Although it has limitations similar to the random oracle model, since in real-world scenarios the underlying group structure is usually known and can be exploited, the GGM is widely used for proving the security of cryptographic schemes, such as (EC)DSA [FKP16] or Structure-Preserving Signatures [FHS19].

2.5 Cryptographic Building Blocks

The cryptographic literature is extensive with many well-studied primitives, some of which are used as building blocks to our work. In this section, we present the key primitives used in the following chapters and detail their individual properties.

Signature Schemes. Digital signatures are the analogue of paper signatures, attesting that a message was issued by a given person or entity. They are working in the same way as hand-written signatures, with the same expected properties. Unlike hand-written signatures, unforgeability is attained based on computational problems, and therefore

offers more guarantees than pen-and-paper signatures when one can correctly associate digital signatures public keys to the entiies holding them.

Definition 13: Signature Schemes [PS00]

A signature scheme S is a set of algorithms composed of:

$\mathsf{Gen}_\mathsf{S}(1^\lambda)$**:** is a PPT algorithm which takes as input a security parameter λ and outputs a key pair $(\mathsf{pk}, \mathsf{sk})$.

$\mathsf{Sign}_\mathsf{S}(\mathsf{sk}, m)$**:** is a PPT algorithm which takes on input a message m and a secret key sk and outputs a signature σ.

$\mathsf{Verif}_\mathsf{S}(\mathsf{pk}, m, \sigma)$**:** is a deterministic polynomial-time algorithm, which takes as input a public key pk, a message m, a signature σ and outputs either 0 or 1.

We require that S meets *Correctness* and EUF-CMA as defined below.

Correctness. Ensures the validity of an honestly produced signature. For all $\lambda \in \mathbb{N}$, for all $m \in M$ and for all $(\mathsf{sk}, \mathsf{pk}) \in [\mathsf{Gen}_\mathsf{S}(1^\lambda)]$, we have $\mathsf{Verif}_\mathsf{S}(\mathsf{pk}, m, \mathsf{Sign}_\mathsf{S}(\mathsf{sk}, m)) = 1$.

EUF-CMA (Existantial Unforgeability under adaptative Chosen-Message Attacks)**.** The property ensures that only the person holding the secret key can generate a valid signature for the associated public key. For all PPT algorithms $\mathcal{A}$, there exists a negligible function ϵ such that for any security parameter λ, the following holds $\mathsf{Adv}^{\mathsf{EUF\text{-}CMA}}_{\mathcal{A},\mathsf{S}}(1^\lambda) = \Pr[\mathsf{Exp}^{\mathsf{EUF\text{-}CMA}}_{\mathcal{A},\mathsf{S}}(1^\lambda) = 1] \leq \epsilon(1^\lambda)$ where $\mathsf{Exp}^{\mathsf{EUF\text{-}CMA}}_{\mathcal{A},\mathsf{S}}(1^\lambda)$ is as follows.

$\mathsf{Exp}^{\mathsf{EUF\text{-}CMA}}_{\mathcal{A},\mathsf{S}}(1^\lambda)$	Oracle $\mathcal{O}_{\mathsf{Sign}}(\mathsf{sk}, m)$
1 : $\mathsf{Out} = \emptyset$	1 : $\mathsf{Out} \leftarrow \mathsf{Out} \cup \{m\}$
2 : $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}_\mathsf{S}(1^\lambda)$	2 : $\sigma \leftarrow \mathsf{Sign}_\mathsf{S}(\mathsf{sk}, m)$
3 : $(m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{Sign}_\mathsf{S}(\mathsf{sk},\cdot)}}(\mathsf{pk})$	3 : **return** σ
4 : $b \leftarrow \mathsf{Verif}_\mathsf{S}(\mathsf{pk}, m^*, \sigma^*)$	
5 : **if** $m^* \in \mathsf{Out}$: **return** 0	
6 : **else return** 1	

Asymmetric Encryption Scheme. Data confidentiality has been one of the main concerns of cryptography, even before authentication. *Asymmetric encryption* makes it possible to send an encrypted message on the basis of public knowledge (the public key) and its decryption is only possible for the entity holding the associated private knowledge (the private key).

Definition 14: Asymmetric Encryption [GM84]

An *asymmetric encryption scheme* $\mathcal{E}$ is a set of PPT algorithms composed of:

$\mathsf{Gen}_\mathsf{Enc}(1^\lambda)$**:** is a PPT algorithm which takes on input a security parameter λ, outputs a key pair $(\mathsf{pk}, \mathsf{sk})$.

$\mathsf{Enc}(\mathsf{pk}, p)$**:** is a PPT algorithm which computes and outputs a *ciphertext* c on the message m using the public key pk.

$\mathsf{Dec}(\mathsf{sk}, c)$**:** is a deterministic polynomial time algorithm, which takes as input a secret key sk, and a ciphertext c and outputs a *plaintext* p.

We require that $\mathcal{E}$ meets *Correctness* and IND-CCA as defined below.

Correctness. For all message m, for all security parameter $\lambda \in \mathbb{N}$, for all $(\mathsf{sk}, \mathsf{pk}) \in [\mathsf{Gen}_{\mathsf{Enc}}(1^\lambda)]$, we have $\mathsf{Dec}(\mathsf{sk}, \mathsf{Enc}(\mathsf{pk}, m)) = m$.

IND-CCA (Indistinguishable under adaptative Chosen Ciphertext Attack)**.** Guarantees that only the person holding the secret key can distinguish a ciphertext from a random element, even if access to other choosen decrypted messages is provided. For all PPT adversaries $\mathcal{A}$ which is provided a decryption oracle $\mathcal{O}_{\mathsf{Dec}(\mathsf{sk},\cdot)}$ (which rejects the challenge c when inputted), the following quantity is at most negligible.

$$\left| \Pr \left[\begin{array}{l} (\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{Gen}_{\mathsf{Enc}}(1^\lambda), (m_0, m_1) \leftarrow \mathcal{A}_1^{\mathcal{O}_{\mathsf{Dec}(\mathsf{sk},\cdot)}}(\mathsf{pk}) \\ b \xleftarrow{\$} \{0,1\}, c \leftarrow \mathsf{Enc}(\mathsf{pk}, m_b), b^* \leftarrow \mathcal{A}_2^{\mathcal{O}_{\mathsf{Dec}(\mathsf{sk},\cdot)}}(c) \end{array} : \; b = b^* \right] - \frac{1}{2} \right|$$

Other levels of security for asymetric encryption schemes exist, for example IND-CPA, which is equivalently defined but this time without the decryption oracle access being given to the adversaries algorithms $\mathcal{A}_1$ and $\mathcal{A}_2$. Existing models are defined and compared in [WSI02]. We only introduced the ones which are required to ensure the properties of our constructions.

Classical example of a asymetric encryption is Elgamal encryption [ElG85], defined for a group $\mathbb{G}$ of prime order p with a generator g, by the following algorithms:

$\mathsf{Gen}_{\mathsf{Enc}}(1^\lambda)$**:** picks $\mathsf{sk} \xleftarrow{\$} \mathbb{Z}_p^*$ and computes $\mathsf{pk} = g^{\mathsf{sk}}$. Returns $(\mathsf{pk}, \mathsf{sk})$.

$\mathsf{Enc}(\mathsf{pk}, m)$**:** draws $y \xleftarrow{\$} \mathbb{Z}_p^*$ and returns $c = (c_1 = g^y, c_2 = m \cdot \mathsf{pk}^y)$.

$\mathsf{Dec}(\mathsf{sk}, c)$**:** parses c as (c_1, c_2) and returns $m = c_2 \cdot c_1^{-\mathsf{sk}}$.

Elgamal is *partially homomorphic*, *i.e.,* there exists an efficient operation $\cdot$ such that $\mathsf{Enc}(\mathsf{pk}, m) \cdot \mathsf{Enc}(\mathsf{pk}, m') = \mathsf{Enc}(\mathsf{pk}, m \cdot m')$, and *randomizable*, *i.e.,* there exists an efficient algorithm Rand that changes a ciphertext c into a new ciphertext c' of the same plaintext:

$\mathsf{Rand}(c, r, \mathsf{pk})$: parses c as (c_1, c_2) and returns $c' = (c_1' = c_1 \cdot g^r, c_2' = c_2 \cdot \mathsf{pk}^r)$.

Property 1

The Elgamal encryption scheme described above is IND-CPA secure under the DDH assumption.

Elgamal encryption does not guarantee IND-CCA security as defined above, as it is a randomisable encryption scheme. Another reason is that Elgamal encryption is homomorphic, which also prevents IND-CCA security from being achieved. It has recently been shown that Elgamal encryption does not achieve even the non-adaptive variant of IND-CCA [Sch24], in which the adversary $\mathcal{A}_2$ does not have access to a decryption oracle $\mathsf{Dec}(sk, \cdot)$, hence cannot make decryption requests after receiving the challenge c.

Commitment. A Commitment allows one party (the committer) to commit to a value while keeping it hidden. The commiter then has the ability to reveal the committed value whenever it is needed.

Definition 15: Commitment

A *commitment* is a tuple of PPT algorithms:

$\mathsf{Setup}(1^\lambda)$ is a PPT algorithm that outputs parameters pp containing the definition of the commitment space.

$\mathsf{Commit}(x, r)$ is a PPT algorithm that, on input a message x and some randomness r in a set R outputs a commitment c.

$\mathsf{Open}(x, c, r)$ is a PPT algorithm that, on input a message x, and values r and c outputs either 0 or 1.

We require that a commitment meets the *Hiding* and *Binding* properties as defined below.

Prefect Hiding. The *Hiding* property ensures that the commitment c does not reveal any information about the committed value x. For all PPT adversaries $\mathcal{A}$ it is expected that:

$$\left| \Pr \left[\begin{array}{l} \mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda), (x_0, x_1) \leftarrow \mathcal{A}(\mathsf{pp}) \\ b \xleftarrow{\$} \{0,1\}, r \xleftarrow{\$} R, c \leftarrow \mathsf{Commit}(x_b, r), b^* \leftarrow \mathcal{A}(c) \end{array} : \; b = b^* \right] - \frac{1}{2} \right| = 0.$$

Computationally Binding. The *binding* property ensures that once the committer has chosen a value x and created a commitment c, it cannot open the commitment with a value x' that would be different fro the first one. For all PPT adversaries $\mathcal{A}$ the following probability is at most negligible,

$$\Pr \left[\begin{array}{l} \mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda), \\ (x, x', r, r', c) \leftarrow \mathcal{A}(\mathsf{pp}) \end{array} : \begin{array}{l} \mathsf{Open}(x, c, r) = \mathsf{Open}(x', c, r') \\ x \neq x' \end{array} \right] \leq \epsilon(1^\lambda).$$

Structure-Preserving Signature. Full named *Structure-Preserving Signatures for Equivalence Classes*, these are signatures on mathematically structured messages. Originally issued for a message, the signature can be freely (without the usage of any secret key) adapted to any given element in the equivalence class of the original messages. In addition to the classic properties of correctness and unforgeability (EUF-CMA) that apply to any signature scheme, the adaptation process must be ambiguous: distinguishing whether the signature has been updated with the message or whether a new one has been produced based on a new message, should be unfeasible.

Definition 16: Structure-Preserving Signatures [FHS19]

A *Structure-Preserving Signatures for Equivalence Classes* SPS for an equivalence classe $\mathcal{R}$ over a group $\mathbb{G}$ is a set of algorithms composed of:

$\mathsf{Gen}_{\mathsf{SPS}}(1^\lambda, l; \mathcal{R})$**:** is a PPT algorithm which takes as input a security parameter λ and an integer integer $l > 1$. It returns a key pair $(\mathsf{pk}, \mathsf{sk})$.

$\mathsf{Sign}_{\mathsf{SPS}}(\mathsf{sk}, m; \mathcal{R})$**:** is a PPT algorithm which takes as input a secret key sk and a message m. It returns a signature σ.

$\mathsf{ChgRep}_{\mathsf{SPS}}(m, \sigma, \mu, \mathsf{pk}; \mathcal{R})$**:** is a PPT algorithm which takes as input a representative m of an equivalence class, a signature σ, a scalar μ, and a public key pk. It returns an updated signature σ' for the message m^μ.

$\mathsf{Verif}_{\mathsf{SPS}}(m, \sigma, \mathsf{pk}; \mathcal{R})$**:** is a deterministic polynomial-time algorithm, which takes as input a public key pk, a message m, a signature σ. It returns either 0 or 1.

We require that SPS meets Correctness, EUF-CMA and *Signature Adaptation* as defined below.

Correctness. Let $\mathcal{R}$ be a relation. For all $l \in \mathbb{N}$, for all security parameters λ, $(\mathsf{pk}, \mathsf{sk}) \in [\mathsf{Gen}_{\mathsf{SPS}}(1^\lambda, l; \mathcal{R})]$, $m \in \mathbb{G}^l$, and $\mu \in \mathbb{Z}_p^*$, the following equations should be true : $\mathsf{Verif}_{\mathsf{SPS}}(m, \mathsf{Sign}_{\mathsf{SPS}}(\mathsf{sk}, m; \mathcal{R}), \mathsf{pk}; \mathcal{R}) = 1$ and

$$\mathsf{Verif}_{\mathsf{SPS}}(\mu m, \mathsf{ChgRep}_{\mathsf{SPS}}(m, \mathsf{Sign}_{\mathsf{SPS}}(\mathsf{sk}, m; \mathcal{R}), \mu, \mathsf{pk}; \mathcal{R}), \mathsf{pk}; \mathcal{R}) = 1.$$

EUF-CMA (Existantial Unforgeability under adaptative Chosen-Message Attacks)**.** Let $l > 1$ and λ a given security parameter. The probability,

$$\Pr\left[\begin{matrix}(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}_{\mathsf{SPS}}(1^\lambda, l; \mathcal{R}), \\ (m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathsf{Sign}_{\mathsf{SPS}}(\cdot, \mathsf{sk}; \mathcal{R})}(\mathsf{pk}, l)\end{matrix} : \begin{matrix}\forall m \in \mathcal{S}, m^* \notin [m]_{\mathcal{R}} \wedge \\ \mathsf{Verif}_{\mathsf{SPS}}(m, \sigma, \mathsf{pk}; \mathcal{R})\end{matrix}\right]$$

should be negligible for every PPT adversary $\mathcal{A}$, where $\mathcal{S}$ is the set of message queries that $\mathcal{A}$ has issued to the signing oracle.

Signature Adaptation. Randomised signatures must be indistinguishable from new signatures. Let $l > 1$ and λ the security parameter, and consider any key pair $(\mathsf{pk}, \mathsf{sk}) \in [\mathsf{Gen}_{\mathsf{SPS}}(1^\lambda, l; \mathcal{R}))]$, $\mu \in \mathbb{Z}_p^*$ and $m \in \mathbb{G}^l$. For all tuples $(\mathsf{sk}, \mathsf{pk}, m, \sigma, \mu)$ the distributions of the signature algorithm $\mathsf{Sign}_{\mathsf{SPS}}(\mathsf{sk}, m; \mathcal{R})$ and an update signature generation algorithm $\mathsf{ChgRep}_{\mathsf{SPS}}(m, \sigma, \mu, \mathsf{pk}; \mathcal{R})$ are identical.

Throughout this manuscript, we set the relation $\mathcal{R}$ over a group $\mathbb{G}$ to $\mathcal{R} = \{(m, m') \in \mathbb{G}^l \times \mathbb{G}^l \mid \exists \mu \in \mathbb{Z}_p, m' = m^\mu\}$ defining equivalence classes $[M]_{\mathcal{R}} \subset \mathbb{G}$ for all elements $M \in \mathbb{G}^l$. As we have established and fixed the relation $\mathcal{R}$, we will no longer explicitly write it when referring to the algorithms of an SPS scheme.

We have designed our models and the signature introduced in Chapter 4 with the Structure-Preserving Signatures for Equivalence Classes introduced by Fuchsbauer *et al.* [FHS19] in mind. The scheme details can be found in Figure 2.1 on the next page. In [FHS19], key generation involves the creation of l elements in the keys enabling the signing of messages of dimension l over $\mathbb{G}$ (*i.e.,* $m \in \mathbb{G}^l$). With a public key designed for dimension l signatures, it is possible to sign lower-dimensional messages using a subset of the generated elements and without security loss. We rely on this to instantiate signatures for vectors with less than l elements.

$\underline{\mathsf{Gen}_{\mathsf{SPS}}(1^\lambda, l; \mathcal{R})}$**:** chooses $\boldsymbol{x} \in (\mathbb{Z}_p^*)^l$ and sets $\mathsf{sk} = \boldsymbol{x}$ and $\mathsf{pk} = g_2^{\boldsymbol{x}} = (g_2^{x_1}, \ldots, g_2^{x_l})$.

$\underline{\mathsf{Sign}_{\mathsf{SPS}}(\mathsf{sk}, m; \mathcal{R})}$**:** parses $\mathsf{sk} = \boldsymbol{x}$, $m \in (\mathbb{G}_1^*)^l$, chooses $y \xleftarrow{\$} \mathbb{Z}_p$ and computes

$$Z_1 = \left(\prod_{i=1}^{l} m_i^{x_i}\right)^y \quad ; \quad Y_1 = g_1^{\frac{1}{y}} \quad ; \quad Y_2 = g_2^{\frac{1}{y}}.$$

Returns $\sigma = (Z_1, Y_1, Y_2)$.

$\underline{\mathsf{ChgRep}_{\mathsf{SPS}}(m, \sigma, \mu, \mathsf{pk}; \mathcal{R})}$**:** verifies the signature, chooses $\psi \xleftarrow{\$} \mathbb{Z}_p^*$, and returns

$$(Z_1^{\psi\mu}, Y_1^{\frac{1}{\psi}}, Y_2^{\frac{1}{\psi}}).$$

$\underline{\mathsf{Verif}_{\mathsf{SPS}}(m, \sigma, \mathsf{pk}; \mathcal{R})}$**:** parses $\mathsf{pk} = (\mathsf{pk}_1, \ldots, \mathsf{pk}_l)$, $m \in (\mathbb{G}_1^*)^l$ and returns 1 if the following holds and 0 otherwise:

$$\prod_{i=1}^{l} e(m_i, \mathsf{pk}_i) = e(Z_1, Y_2) \quad \wedge \quad e(Y_1, g_2) = e(g_1, Y_2).$$

Figure 2.1: SPS Signature Scheme from [FHS19].

Secret Sharing. Secret Sharing is used to distribute a secret or a piece of sensitive information to a group of participants in such a way that only specific subsets of participants can recover the secret. The basic principle is to divide the secret into shares or parts, which are distributed amongst the participants. These parts do not individually reveal any information about the secret, but when all parts are combined, the original secret can be reconstructed. We present here a simple model where all part are required to recover the secret.

Definition 17: Secret Sharing [Sha79]

A *secret sharing* scheme SS amongst n participants is given by:

$\mathsf{Split}(m, n) \to (s_i)_{1 \le i \le n}$**:** is a PPT algorithm that takes as parameters n and a message m and returns a vector of shares $(s_i)_{1 \le i \le n}$.

$\mathsf{Recover}((s_i)_{1 \le i \le n}) \to m$**:** is a deterministic polynomial-time algorithm that takes a vector of shares $(s_i)_{1 \le i \le n}$ and returns a message m.

It must verify the *correctness* described by the equality $\mathsf{Recover}(\mathsf{Split}(m, n)) = m$ and achieve *Perfect Secrecy* as defined below.

Perfect Secrecy. Recovering a message m split for a threshold of k with less than k shares is unfeasible. The experiment $\mathsf{Exp}^{\mathsf{perf\text{-}Sec}}_{\mathcal{A},\mathsf{SS}}(1^\lambda)$ for an adversary $\mathcal{A}$ and for a secret sharing scheme is defined as:

$\mathsf{Exp}^{\mathsf{perf\text{-}Sec}}_{\mathcal{A},\mathsf{SS}}(1^\lambda)$ - (Perfect Secrecy)

1 : $m_0, m_1, n, k \leftarrow \mathcal{A}(\lambda)$ // The value k must be contained in the set $\{1, \ldots, n\}$.

2 : $b \xleftarrow{\$} \{0,1\}$

3 : $(s_i)_{1\leq i\leq n} \leftarrow \mathsf{Split}(m_b, n)$ // Split one of the messages based on a uniform distribution.

4 : $b' \leftarrow \mathcal{A}((s_i)_{1\leq i\leq n, i\neq k})$ // Here $\mathcal{A}$ must operate Recover with one less share than necessary.

5 : **return** $(b = b')$

and for any adversary it should lead to:

$$\mathsf{Adv}^{\mathsf{perf\text{-}Sec}}_{\mathcal{A},\mathsf{SS}}(1^\lambda) = \left|\Pr[\mathsf{Exp}^{\mathsf{perf\text{-}Sec}}_{\mathcal{A},\mathsf{SS}}(1^\lambda) = 1] - 1/2\right| = 0.$$

The threshold version in which the minimum number of parts required to retieve the secret can be set to any number do exist. A notable example of this is Shamir's secret sharing scheme [Sha79], which is based on *Lagrange polynomials*.

We remain in our model for secret sharing scheme without threshold and described a scheme according to our Definition 17 in Figure 2.2.

$\mathsf{Split}(m \in \{0,1\}^k, n)$	$\mathsf{Recover}(s_1, \ldots, s_n)$
1 : $s_1, \ldots, s_{n-1} \xleftarrow{\$} \{0,1\}^k$	1 : $m \leftarrow \bigoplus_{i=1}^{n} s_i$
2 : $s_n \leftarrow m \oplus \bigoplus_{i=1}^{n-1} s_i$	2 : **return** m
3 : **return** $(s_i)_{i=1}^n$	

Figure 2.2: Description of the Secret Sharing Scheme.
(For any $k \in \mathbb{N}$ sufficiently large for the inclusion.)

Zero-knowledge Proofs. Our constructions also make use of *Non-Interactive Zero-Knowledge proofs of knowledge* (NIZK) [GMR89]. A *Zero-Knowledge Proof* (ZK) is used to prove knowledge of a secret solution for a public statement by interacting with the verifier. This statement is often a computationaly hard problem. For example, we will present an example of a proof demonstrating knowledge of the discrete logarithm of an element of a group. Interaction in ZK proofs reduces their practicality, often their non-interactive counterpart NIZK proofs are prefered to reduce the overheads associated with additional communication. These primitives must have the following properties:

Perfect-Completeness. If the statement is true, an honest verifier will be convinced at the end of the procedure.

Soundness. If the statement is false, no cheating prover can convince an honest verifier that the statement is true.

Zero-knowledge. The verifier learns nothing other than the fact that the statement is true.

Let $\mathcal{R}$ be a binary relation and ϕ, w two elements verifying $(\phi, w) \in \mathcal{R}$, ϕ is called the *statement* of the relation and w the *witness*. A NIZK is a cryptographic primitive

allowing a prover knowing a witness w that w and ϕ verify the relation $\mathcal{R}$, *i.e.,* $(\phi, w) \in \mathcal{R}$, leaking no information on w. Let $\mathcal{L}_{\mathcal{R}}$ denote the language associated to the binary relation $\mathcal{R}$ defined by $\mathcal{L}_{\mathcal{R}} = \{\phi | \exists w, (w, \phi) \in \mathcal{R}\}$. Let us denote $\mathsf{View}(P, \mathsf{V}(x))$ the view of the messages sent an received by $\mathcal{V}$ when interacting with $\mathcal{P}$ on common input x.

Definition 18: Zero-Knowledge Proof of Knowledge

A *Zero-Knowledge Proof of Knowledge* between a *prover* $\mathcal{P}$ and a *verifier* $\mathcal{V}$ for a family of languages $\mathcal{L}$ with relation $\mathcal{R}$ is any pair of PPT algorithms such that

$\mathsf{Setup}(1^\lambda)$**:** is a PPT algorithm which takes as input a security parameter 1^λ, and returns the public parameters pp which are inplicit inputs of all the other algorithms.

$\mathsf{Prove}(\mathcal{P}(x), \mathcal{V}(\phi))$**:** is a two party protocol between $\mathcal{P}$ and $\mathcal{V}$ with $\mathcal{V}$ outputting a bit $b \in \{0, 1\}$.

Zero-knowledge proof of knowledge must achieve *Completeness*, *Soudness*, *Honest Verifier Zero-Knowledge* and *Extractability* as defined below.

Perfect Completeness. The proof system $(\mathcal{P}, \mathcal{V})$ for a family of languages $\mathcal{L}$ with relation $\mathcal{R}$ satisfies *perfect completeness* if for any statement ϕ with a witness w, it holds that:

$$\Pr[\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda), (\phi, w) \xleftarrow{\$} \mathcal{R} : \mathsf{Prove}(\mathcal{P}(x), \mathcal{V}(\phi)) = 1] = 1.$$

Soundness. A ZK proof system for a family of languages $\mathcal{L}$ with relation $\mathcal{R}$ provides *soundness* if for any outcome of $\mathsf{Setup}(1^\lambda)$, for any $\phi \notin \mathcal{L}_{\mathcal{R}}$, for all unbounded (*resp.* PPT) adversary $\mathcal{A}$:

$$\Pr[\mathsf{Prove}(\mathcal{A}(\mathsf{pp}, \phi), \mathcal{V}(\phi)) = 1] = 0 \ (\textit{resp.} \ \leq \epsilon(1^\lambda)).$$

Honest Verifier Zero-Knowledge. The proof system $(\mathcal{P}, \mathcal{V})$ for a family of languages $\mathcal{L}$ with relation $\mathcal{R}$ satisfies the *Honest Verifier Zero-Knowledge* if there exists a probabilistic simulator Sim running in expected polynomial-time such that for all $\mathsf{pp} \in [\mathsf{Setup}(1^\lambda)]$ and for all statements ϕ, the distribution of $\mathsf{View}(\mathsf{Prove}(\mathcal{P}(w), \mathcal{A}(\phi)))$ and $\mathsf{Sim}(\mathsf{pp}, \phi)$ are indistinguishable. This indistinguishability can either hold computationaly or statistically.

Extractability. The proof system $(\mathcal{P}, \mathcal{V})$ for a family of languages $\mathcal{L}$ with relation $\mathcal{R}$ satisfies *extractability* if there exist a polynomial-time knowledge *extractor* Ext and a negligible function $\epsilon(1^\lambda)$ such that, for all $\mathsf{pp} \in [\mathsf{Setup}(1^\lambda)]$, for all statement ϕ, for any algorithm $\mathcal{A}^{\mathsf{Sim}(\cdot,\cdot)}$ that outputs a fresh statement (s, π) with $\mathsf{Verif}_{\mathsf{ZK}}(s, \pi) = 1$ such that $\mathcal{A}$ has access to a simulator that forges proofs for chosen statements, $\mathsf{Ext}^{\mathcal{A}}$, having access to $\mathcal{A}$, outputs w such that $(s, w) \in \mathcal{R}$ with probability $1 - \epsilon(1^\lambda)$.

Schnorr Proof and Σ-Protocol. The Schnorr zero-knowledge proof, described below, allows a prover to demonstrate knowledge of a secret $x \in \mathbb{Z}_p^*$, a discrete logarithm

value, by only revealing the public information $y = g^x$, for a generator g of a group $\mathbb{G}$ of primer order p. It is an interactive proof of knowledge where the proof protocol takes place in three stages: the prover sends a *commitment* to the verifier, the verifier sends a *challenge* to the prover and the prover sends a final *response* which is *verified* by the verifier. Let x be the prover's witness and $y = g^x$ its statement. The protocol ensuring that the verifier, generating the challenge, and verifying the proof, learns nothing on x beyond the validity of the statement and the value of y.

Commitment: The prover selects a random value $r \in \mathbb{Z}_p^*$ and computes the commitment $C = g^r$.

Challenge: The verifier sends a random challenge $c \in \mathbb{Z}_p^*$ to the prover.

Response: The prover computes $s = r + cx$, where x is the prover's secret and r the random value used for the commitment. The value s is sent back to the verifier.

The verifier then checks the validity of the proof by verifying that

$$g^s \equiv C \cdot y^c.$$

This type of proof are called Σ-*protocols* [Sch91].

Definition 19: Σ-Protocol [Sch91]

Let $\mathcal{L}$ be a family of languages with relation $\mathcal{R}$ and let λ be a security parameter. Let $\mathcal{C}$ be a challenge space of size the security parameter λ. A Σ-*Protocol* Π for a language $\mathcal{L}$ is a 3-move protocol between a prover P and a verifier V consisting of a tuple of algorithms $\Pi = (A, \mathcal{C}, Z)$ with the following interfaces:

$\mathsf{Setup}(1^\lambda)$**:** is a PPT algorithm which takes as input a security parameter λ and returns public parameters pp which are inplicit inputs of all the other algorithms.

$A(\phi, w)$ **(Commitment):** is a PPT algorithm which takes as input a statement ϕ, the corresponding witness w, such that $\mathcal{R}(\phi, w) = 1$, and outputs the first message a that P sends to V in the first round. It also outputs an element r.

$c \xleftarrow{\$} \mathcal{C}$ **(Challenge):** is the sample of a random challenge c, that V sends to P in the second round.

$Z(\phi, w, c, r)$ **(Response):** is a PPT algorithm which takes as input the statement ϕ, the witness w, the challenge c and the element r. Outputs the message z that P sends to V in the third round.

$\mathsf{Verif}_{\mathsf{ZK}}(\phi, a, c, z)$ **(Verification):** is a deterministic polynomial-time algorithm run by V which takes as input the statement ϕ, prover's messages a, z, and the challenge c, this algorithm run by V, outputs a bit $b \in \{0, 1\}$.

We require that the ZK proof system $(\mathsf{Setup}, (A, Z, \mathsf{Verif}_{\mathsf{ZK}}))$ Σ-Protocol guarantees *Completeness*, *Soudness*, *Honest Verifier Zero-Knowledge* and *Extractability* as defined previously.

Non-Interactive Zero-Knowledge Proof. As we have seen, zero-knowledge proofs are systems executed between a prover and a verifier. Of all the existing zero-knowledge proofs, a subset requires no more than one message sent by the prover to the verifier. This message is often referred to as a *proof* and is denoted by π. Below we give a refined definition of a zero-knowledge proof that requires no interaction between the prover and the verifier and for which the verifier (or any verifier) can be convinced only by seeing the statement ϕ and the proof π.

Definition 20: Non-Interactive Zero-Knowledge Proof [GM17]

A *Non-Interactive Zero-Knowledge proof* (NIZK) for a family of languages $\mathcal{L}$ with relation $\mathcal{R}$ is a set of PPT algorithm composed of:

$\mathsf{Setup}(1^\lambda)$**:** is a PPT algorithm which takes as input a security parameter λ and returns the public parameters pp which are inplicitly input to all the other algorithms.

$\mathsf{ZK}(w; \phi)$**:** is a probabilistic polynomial-time process, which takes as input a witness w for a statement ϕ and returns a proof π that $(\phi, w) \in \mathcal{R}$.

$\mathsf{Verif}_{\mathsf{ZK}}(\phi, \pi)$**:** is a deterministic polynomial-time algorithm, which takes as input an instance ϕ and a proof π, and returns either 0 or 1.

We require that NIZK satisfies *completeness*, *soundness*, and *zero-knowledge*, as defined below. ZK proofs may also meet Knowledge-extraction as defined below.

Perfect Completeness (NIZK). A NIZK proof for a family of languages $\mathcal{L}$ with relation $\mathcal{R}$ satisfies *perfect completeness* if given a valid statement, a honest prover with a witness can convince the honest verifier:

$$\Pr[\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda), (\phi, w) \xleftarrow{\$} \mathcal{R}, \pi \leftarrow \mathsf{ZK}(\phi, w) : \mathsf{Verif}_{\mathsf{ZK}}(\phi, \pi) = 1] = 1.$$

Soundness. A NIZK proof for a family of languages $\mathcal{L}$ with relation $\mathcal{R}$ satisfies perfect (*resp.* computational) *soundness* if for any $\mathsf{pp} \in [\mathsf{Setup}(1^\lambda)]$, for any statement $\phi \notin \mathcal{L}_{\mathcal{R}}$, for all unbounded (*resp.* PPT) adversary $\mathcal{A}$, it holds that:

$$\Pr[\pi \leftarrow \mathcal{A}(\mathsf{pp}, \phi) : \mathsf{Verif}_{\mathsf{ZK}}(\phi, \pi) = 1] = 0 \ (\textit{resp.} \ \leq \epsilon(1^\lambda)).$$

Zero-knowledge. A NIZK proof guarantees *zero-knowledge* if it does not disclose any additional information other than the truth of the instance $(\phi, w) \in \mathcal{R}$. This is modeled based on the ability to distinguish between an honestly-generated proof and a simulator generating valid proofs without holding the witness based on a trapdoor information. A NIZK proof for a family of languages $\mathcal{L}$ with relation $\mathcal{R}$ provides perfect (*resp.* computational) *zero-knowledge* if there exists a

probabilistic simulator Sim running in expected polynomial-time such that for all $\mathsf{pp} \in [\mathsf{Setup}(1^\lambda)]$, for all statements ϕ, for all PPT (alternatively unbounded) adversary $\mathcal{A}$, $\mathsf{Adv}^{\mathsf{Sim}}_{\mathcal{A},\mathsf{NIZK}}(1^\lambda) = |\Pr[\mathsf{Exp}^{\mathsf{Sim}}_{\mathcal{A},\mathsf{NIZK}}(1^\lambda)] - 1/2| = 0$ (*resp.* $\leq \epsilon(1^\lambda)$) for $\mathsf{Exp}^{\mathsf{Sim}}_{\mathcal{A},\mathsf{NIZK}}(1^\lambda)$.

$\mathsf{Exp}^{\mathsf{Sim}}_{\mathcal{A},\mathsf{NIZK}}(1^\lambda)$

1 : $b \xleftarrow{\$} \{0,1\}$
2 : **if** $b = 0, \pi \leftarrow \mathsf{ZK}(\mathsf{pp}, \phi, w)$
3 : **if** $b = 1, \pi \leftarrow \mathsf{Sim}(\mathsf{pp}, \phi)$
4 : $b^* \leftarrow \mathcal{A}^{\mathsf{Sim}}(\mathsf{pp}, \phi, \pi)$
5 : **return** $b = b'$

We also sometime require knowledge extractability for NIZK proofs. This property implies the soudness of the proof [Cou17].

Knowledge-extraction. An NIZK scheme is *Knowledge-extractable* if, whenever a prover produces a valid argument, it is possible to extract a valid witness from their state information, *i.e.,* if there exists a PPT algorithm Ext called the *knowledge extractor* such that, for all statements ϕ, for all $\mathsf{pp} \in [\mathsf{Setup}(1^\lambda)]$, for all $\phi \in \mathcal{L}_\mathcal{R}$, for all PPT (alternatively unbounded) adversary $\mathcal{A}$, the advantage $\mathsf{Adv}^{\mathsf{Ext}}_{\mathcal{A},\mathsf{NIZK}}(1^\lambda) = |\Pr[\mathsf{Exp}^{\mathsf{Ext}}_{\mathcal{A},\mathsf{NIZK}}(1^\lambda)] - 1/2| \leq \epsilon(1^\lambda)$ for $\mathsf{Exp}^{\mathsf{Ext}}_{\mathcal{A},\mathsf{NIZK}}(1^\lambda)$ defined as follows.

$\mathsf{Exp}^{\mathsf{Ext}}_{\mathcal{A},\mathsf{NIZK}}(1^\lambda)$

1 : $\pi^* \leftarrow \mathcal{A}^{\mathsf{Sim}}(\mathsf{pp}, \phi)$
2 : $w \leftarrow \mathsf{Ext}(\mathcal{A}(\mathsf{pp}, \phi))$
3 : **return** $\mathsf{Verif}(\phi, \pi^*) \wedge (\phi, w) \notin \mathcal{R}$

We have stated that NIZK proofs are a subset of all ZK proofs. In reality, their exists an efficient transformation, turning most existing ZK proofs into non-interactive ones. This technique is the Fiat–Shamir transformation [FS87] which has been proven secure in the random oracle model [PS00]. Based on this transformation, all sigma zero-knowledge proof protocols can be made non-interactive. Throughout the rest of this thesis, we use the Camenisch and Stadler notation [CS97a], *i.e.,* $\mathsf{ZK}\{w : (w, s) \in \mathcal{R}\}$ denote the proof of knowledge of w for the statement s and the relation $\mathcal{R}$.

Fiat-Shamir Transformation. Now we give the formal definition of the Fiat-Shamir transform, converting any Σ-Protocol into a NIZK by replacing the verifier's challenge with a hash of the commitment and the other state elements. Under the random oracle heuristic this can provide a uniform distribution over the set of challenges.

Definition 21: Fiat-Shamir Transformation [FS87]

Let Π be a Σ-Protocol defined as in Definition 19 on page 41. The *Fiat-Shamir Transformation* of Π is the NIZK proof scheme Π_{NI} defined as follows:

$\mathsf{Setup}(1^\lambda)$**:** generates pp' using the setup algorithm of Π, chooses a hash function $H : \{0,1\}^* \to C$ and returns $\mathsf{pp} = (\mathsf{pp}', H)$.

$\mathsf{ZK}(\phi, w)$ **:** runs $a \leftarrow A(\phi, w)$, $z \leftarrow Z(x, w, c \leftarrow H(\mathcal{R}\|\phi\|a))$ and returns $\pi = (a, z)$.

$\mathsf{Verif}_{\mathsf{ZK}}(\phi, \pi)$**:** parses $\pi = (a, z)$, runs and returns $\mathsf{Verif}_{\mathsf{ZK}}(\phi, a, H(\mathcal{R}\|\phi\|a), z)$.

Property 2: Security of the Fiat-Shamir transform

Let Π be a Σ-Protocol that is complete, sound, honest verifier zero-knowledge and extractable, then the Fiat-Shamir transformation of Π is a NIZK proof scheme that is complete, sound, zero-knowledge and extractable.

Proof. The proof of this property is given in [FS87] □

Signatures of Knowledge. A *Signature of Knowledge* (SoK) is similar to a NIZK except that the proof algorithm $\mathsf{SoK}_m\{w : (w, \phi) \in \mathcal{R}\}$ defined similarly to $\mathsf{ZK}\{w : (w, \phi) \in \mathcal{R}\}$ (also written as $\mathsf{ZK}(\phi, w)$) takes a message m as an additional parameter. As a consequence, the *knowledge-extraction* of a SoK, similar to knowledge-extraction of NIZK proofs, implies that it can be used as an EUF-CMA signature scheme, where ϕ is the public key, w is the secret key, m is the signed message, and π is the signature. We now give the formal definition of its algorithms.

Definition 22: Signature of Knowledge [GM17]

A *Signature of Knowledge* (SoK) for a familly of languages $\mathcal{L}$, for a relation $\mathcal{R}$ and a message space $\mathcal{M}$ is a set of PPT algorithms composed of:

$\mathsf{Setup}_{\mathsf{SoK}}(1^\lambda, \mathcal{R})$**:** is a PPT algorithm which takes as input a relation $\mathcal{R}$ and returns public parameters pp.

$\mathsf{SoK}_m(w; \phi)$**:** is a PPT algorithm which takes as input a pair $(\phi, w) \in \mathcal{R}$ and a message $m \in \mathcal{M}$, and returns a signature σ.

$\mathsf{Verif}_{\mathsf{SoK}}(\phi, m, \sigma)$**:** is a deterministic polynomial-time algorithm which takes as input an instance ϕ, a message m, and a signature σ and outputs 0 or 1.

We require that SoK satisfies *correctness*, *zero-knowledge*, and *soundness* as defined below. SoK may also meet Knowledge-extraction as defined below.

We now present the security model of signatures of knowledge, which is based on the security model for NIZK proofs. Indeed, as we shall see, the Fiat-Shamir framework can be adapted to provide SoK from a ZK proof instead of NIZK. The security requirements

for SoK include only an additional message m in the proof algorithm (called signing algorithm for signatures of knowledge) and the verification algorithm.

Correctness. Analogous to Perfect completeness of NIZK. A signature of knowledge is *correct* if, for a given relation $\mathcal{R}$, $\forall\lambda \in \mathbb{N}$, $\forall(\phi, w) \in \mathcal{R}$, $\forall m \in M$, $\forall \mathsf{pp} \in [\mathsf{Setup}_{\mathsf{SoK}}(1^\lambda, \mathcal{R})]$, $\forall\sigma \in [\mathsf{SoK}_m(w; \phi)]$, it holds that $\Pr[\mathsf{Verif}_{\mathsf{SoK}}(\phi, m, \sigma) = 1] = 1$.

Soundness. A SoK proof system for a relation $\mathcal{R}$ satisfies the *soundness* property if, for any $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$, for every statement $\phi \notin \mathcal{L}_\mathcal{R}$, for all unbounded (*resp.* PPT) adversary $\mathcal{A}$,

$$\Pr[(m^*, \sigma^*) \leftarrow \mathcal{A}(\mathsf{pp}, \phi) : \mathsf{Verif}_{\mathsf{SoK}}(\phi, m^*, \sigma^*)] = 0 \ (\textit{resp.} \ \leq \epsilon(1^\lambda)).$$

Zero-knowledge. A SoK has perfect (*resp.* computational) *zero-knowledge* if it does not disclose any additional information other than the truth of the instance $(\phi, w) \in \mathcal{R}$. This is modeled based on the ability to distinguish between an honestly generated proof and a simulator generating valid proof without holding the witness, solely relying on a trapdoor information. A SoK is *zero-knowledge* if there exists a probabilistic simulator Sim running in expected polynomial-time such that, for all $\mathsf{pp} \in [\mathsf{Setup}(1^\lambda)]$, for all statements ϕ, for all PPT (alternatively unbounded) adversaries $\mathcal{A}$, it holds that $\mathsf{Adv}^{\mathsf{Sim}}_{\mathcal{A},\mathsf{NIZK}}(1^\lambda) = |\Pr[\mathsf{Exp}^{\mathsf{Sim}}_{\mathcal{A},\mathsf{NIZK}}(1^\lambda)]| \leq \epsilon(1^\lambda)$ (*resp.* $\leq \epsilon(1^\lambda)$) for $\mathsf{Exp}^{\mathsf{Sim}}_{\mathcal{A},\mathsf{NIZK}}(1^\lambda)$ define as follows for any $(\phi, w) \in \mathcal{R}$ and for any $m \leftarrow \mathcal{M}$:

$\mathsf{Exp}^{\mathsf{Sim}}_{\mathcal{A},\mathsf{SoK}}(1^\lambda)$

1 : $b \xleftarrow{\$} \{0, 1\}$
2 : **if** $b = 0, \sigma \leftarrow \mathsf{SoK}_m(w; \phi)$
3 : **if** $b = 1, \sigma \leftarrow \mathsf{Sim}(\mathsf{pp}, m, \phi)$
4 : $b^* \leftarrow \mathcal{A}^{\mathsf{Sim}}(\mathsf{pp}, \phi, \sigma)$
5 : **return** $b = b'$

We also sometimes require *knowledge extractability* for SoK proofs. As in the case of NIZK, knowledge extractability may also imply soudness.

Knowledge-extraction. A SoK scheme is *Knowledge-extractable* if whenever a valid argument is produced it is possible to extract a valid witness from their state information, *i.e.,* there exists a PPT algorithm Ext called the *knowledge extractor* such that, for all $\mathsf{pp} \in [\mathsf{Setup}(1^\lambda)]$, for all statement ϕ, for all PPT (alternatively unbounded) adversary $\mathcal{A}$, for all $\phi \in \mathcal{L}_\mathcal{R}$, it holds that $\mathsf{Adv}^{\mathsf{Ext}}_{\mathcal{A},\mathsf{NIZK}}(1^\lambda) = \Pr[\mathsf{Exp}^{\mathsf{Ext}}_{\mathcal{A},\mathsf{NIZK}}(1^\lambda)] \leq \epsilon(1^\lambda)$ for $\mathsf{Exp}^{\mathsf{Ext}}_{\mathcal{A},\mathsf{NIZK}}(1^\lambda)$.

$\mathsf{Exp}^{\mathsf{Ext}}_{\mathcal{A},\mathsf{SoK}}(1^\lambda)$

1 : $(m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathsf{Sim}}(\mathsf{pp}, \phi)$
2 : $w \leftarrow \mathsf{Ext}(\mathcal{A}(\mathsf{pp}, m, \phi))$
3 : **return** $\mathsf{Verif}_{\mathsf{SoK}}(\phi, m, \sigma^*) \wedge (\phi, w) \notin \mathcal{R}$

The Fiat-Shamir framework provides a practical construction for converting any Σ-Protocol into a NIZK proof. It is also possible to convert a Σ-Protocol into a signature of knowledge [CS97b]. In Definition 21 on page 44, instead of generating the challenge as $H(\mathcal{R}\|\phi\|a)$, we embed a message $m \in \mathcal{M}$ in the hash function, then compute $c \leftarrow H(\mathcal{R}\|\phi\|a\|m)$ for the verifier's challenge. This directly extends the transformation and embeds the message into the proof without affecting the security of the NIZK, which becomes a SoK. In fact, the two security models are closely related and differ only in the message inclusion within the proof.

Chapter 3
Modeling Anonymity of Linkable Ring Signatures

Chapter Summary

In this chapter, we point out the imprecise modeling of linkable ring signatures in 16 out of 18 schemes proposed in the literature. We highlight the inability to guarantee the expected anonymity properties as is and identify discrepancies between theoretical models and practical security needs. In light of this problem, we present a refined model that better matches real-world expectations. Here are our four main contributions:

- Model discrepancy: highlighting that current anonymity models for linkable ring signatures fail to provide guarantees on the hiding of user identities for more than one signature.
- Refined security: highlighting the model proposed by Backes *et al.* [BDH+19] for LRS, which corresponds to practical needs but has been ignored in all subsequent work.
- Practical alignment: showing that existing systems are implicitly designed for this improved model and, for most of them, achieve this level of security.
- Classification: providing a complete hierarchy of the two existing anonymity formalisms in the possible corruption models.

Contents

3.1 Introduction to the Chapter Content

Ring signatures [RST01], digital signatures on behalf of *ad hoc* groups hiding which of the entities created them, are amongst the most studied privacy-preserving signatures. Over the years, they have been used in many real-world applications, making them one, if not the most widely deployed type of privacy-preserving signatures. Their applications are numerous and include blockchains (Monero, based on CryptoNote [VS13]), electronic voting [TW05], attestation [TW05], *etc.* These applications regularly require a mitigation of the powerful property of anonymity brought by the original concept.

Anonymity Mitigation. To adapt to its use cases, variations of the original concept have been developed to mitigate its full anonymity. These mitigations, introduced as new properties, are, amongst others, *traceability* of the signer if it produces more than one signature [FS07a], *repudiation* of the signature for non-signers or *claimability* for signers [PS19] and *revocability* of the signer's anonymity by a revocation authority [ZLS+20]. In this chapter, we focus on yet another property: *linkability* of the signature produced by the same signer, and its implications on anonymity. Introduced by Liu *et al.* [LWW04], linkable ring signatures (LRS) have been the subject of many research papers and allows any verifier to link signatures produced by the same signer while concealing the signer's identity under the names of the ring members. A list of existing works is provided in Table 3.1 on the next page. We give an example to illustrate its application and functionality: consider the certification of ballot in an election. Here, each voter signs its ballot paper not only under its identity but also under the identities of all the voters, which allows him to sign its ballot paper without disclosing its identity. This is done by generating a ring signature. In this case, linkable ring signatures would allow an auditor to link the signatures of two electronic ballots from the same entity. This prevents voters from voting multiple times without manipulating the identity of voters, and allows voting to be modified during the elections (as in the Estonian electronic voting system [Val24]).

In the definition of linkable ring signatures, just like ring signatures, include a key generation algorithm, a signature algorithm, and a verification algorithm. Unlike traditional ring signatures, they allow for the verification of whether two signatures were produced by the same signer based on a linking algorithm, while still concealing the signer's identity. This preservation of privacy for the signer is often referred to as *pseudonymity*, *partial anonymity*, or *anonymity*. In the existing literature, the term *anonymity* has been preferred, but we highlight that for linkable ring signatures it represents a weaker property than when applied to ring signatures.

Security Considerations. Four security properties have been defined to model what is expected from linkable ring signatures:

Unforgeability of signatures: it is computationally unfeasible for anyone who is not part of the ring to produce a valid signature that would be accepted as legitimate.

Anonymity of signer: given a signature, it is unfeasible to determine which member of the ring generated it.

Reference	Assumption	Model
Liu *et al.* [LWW04]	DL related	ROM
Tsang *et al.* [TWC+05]	Strong RSA & DDH	ROM
Liu and Wong [LW05]	DL related	ROM
Tsang and Wei [TW05]	DL related	ROM
Liu *et al.* [LASZ13]	DL related	ROM
Yuen *et al.* [YLA+13]	DL related	Standard
Boyen and Haines [BH18]	CDL[1]	ROM
Branco and Mateus [BM18]	GSDD[2]	ROM
Baum *et al.* [BLO18]	SIS, LWE	ROM
Lu *et al.* [LAZ19]	SIS	ROM
Liu *et al.* [LNY+19]	M-SIS, D-MLWE	ROM
Zhang *et al.* [ZLS+20]	DL related	ROM
Balla *et al.* [BBG+22]	DL related	ROM
Bootle *et al.* [BEHM22]	DL related	ROM
Xiangyu *et al.* [HC24]	DL related	ROM
Xue *et al.* [XLAZ24]	Generic construction	ROM

(a) Existing Linkable Ring Signatures Proven Secure Under The Model for One-time Anonymity 1-ano.

Reference	Assumption	Model
Alberto *et al.* [ATSS+18]	R-SIS	ROM

(b) Existing One-time Linkable Ring Signatures.

Reference	Assumption	Model
Backes *et al.* [BDH+19]	Generic construction	Standard
Beullens *et al.* [BKP20]	SIDH, M-LWE	ROM

(c) Existing Linkable Ring Signatures with Proven Anonymity ano.

Table 3.1: Existing Linkable Ring Signatures.

Linkability of signatures: it is unfeasible to generate two unlinked signatures from the same secret key.

Non-slanderability of signatures: it is unfeasible to create a situation where a valid signature is falsely claimed to be generated by another member of the ring.

Of these properties, *unforgeability* and *anonymity* are derived from ring signatures, while the other two are necessary to guarantee the security of the linkability. Although supposedly adapted from ring signatures, the level of anonymity formalised by most previous works, even the most recent ones, is insufficient. In fact, the associated constructions could suffer from a total lack of anonymity. What is more, the environments in which they could suffer concrete breaches in the anonymity of entities. In recent works such as [BEHM22] and the other schemes of Table 3.1 (except for [ATSS+18]), *Anonymity* (ano) is informally characterised by the following statement:

> *"Anonymity, demands that an adversary cannot tell which of a ring's secret keys was used to produce a signature."*

Despite the accurate informal descriptions, we show in this chapter that the definitions for all schemes listed in Table 3.1a essentially formalise this same concept as follows:

[1] CDL: Central Decoding Problem
[2] GSD: General Syndrome Decoding

> *Anonymity demands that an adversary cannot tell which of a ring's secret keys was used to produce* ***an entity's first signature****.*

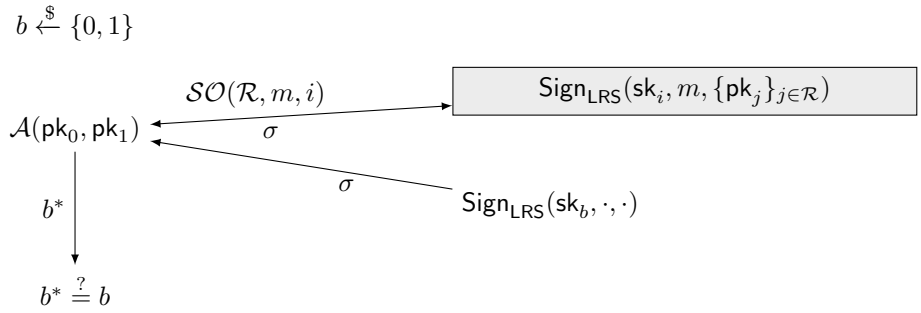
We see a direct implication of the second statement by the first one. Throughout this chapter, we refer to the second quote and weaker notion as *One-time Anonymity* (1-ano).

And, while it may be a feature of some schemes, as in [TW05], which main characteristics are described in Table 3.1b, this statement does not model the actual expectation formulated for linkable ring signatures in the literature. Figure 3.1 on the next page shows a schematic comparison of the experiment of anonymity of ring signatures and the most frequently used one-time anonymity (1-ano) of linkable ring signatures. In Figure 3.1b, the one depicting the anonymity of linkable ring signatures, there is no guarantee regarding what the second signature might reveal about the identity of the signer, as we elucidate below. This is why we refer to this definitions of anonymity as *one-time anonymity* in order to better reflect the actual guarantees provided by the formalisation of this property. In looking for the rationale behind such a definition, one might speculate that it is linked to a statement made in Bender *et al.*'s seminal paper [BKM06], whose provided a security framework for ring signatures. The statement in question is as follows:

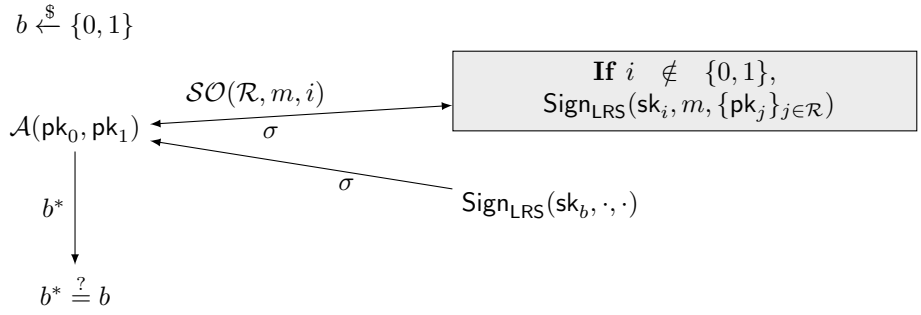
> *"a weaker definition of anonymity* (one-time anonymity of Figure 3.1b) *whereby the adversary obtains only users' public keys and a single signature – but cannot obtain multiple other signatures via a signing oracle – does not imply unlinkability [of the signatures produced by the same signer]" .*

At first glance, removing the right to obtain multiple signatures in the experiment may seem like a reasonable way of defining anonymity with linkability. However, upon closer examination, this statement actually discusses the fact that unlinkability is not considered when only one signature is issued to the adversary. Therefore, all we can ascertain about the definition of anonymity is that this weak definition of one-time anonymity 1-ano appeared in the very first articles on linkable ring signatures and has persisted across most existing schemes (of Table 3.1a). Only two existing works [BDH+19, BKP20], reported in Table 3.1c on the preceding page have formalised the anonymity of LRS in a more realistic experiment, schematically described in Figure 3.1c on the next page using a *Left or Right* challenge oracle. However, their model is left unconsidered in all subsequent works reported in Table 3.1a.

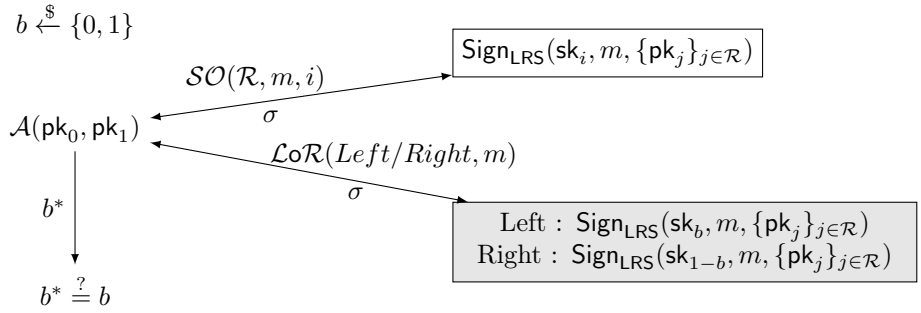
Our Contributions. In this chapter, we argue that the modelisation of the anonymity experiment for linkable ring signatures in all the schemes cited in Table 3.1a does not match the security expectations formalised in their respective works. This discrepancy means that 16 of the 18 existing linkable ring signatures may suffer from a deep lack of protection of the signer's identity after only the second signature. The most commonly used security model for linkable ring signatures, which we have referred to as one-time anonymity (1-ano) (above and in Figure 3.1b on the facing page), remains broadly similar across all the works listed in Table 3.1a. The one-time anonymity experiment only hides the identity of the signer when they first sign, not necessarily on the second signature. This does not match the informal expectations described in all these works. Our main



(a) Anonymity of Ring Signatures [BKM06].



(b) One-time Anonymity 1-ano of Linkable Ring Signatures ($\mathsf{Exp}^{\text{1-ano}}_{\mathsf{LRS}}(1^\lambda)$ in Section 3.2). Referred to as *Anonymity* in all the articles cited in the Table 3.1a.



(c) Anonymity ano of Linkable Ring Signatures ($\mathsf{Exp}^{\mathsf{ano}}_{\mathcal{A},\mathsf{LRS}}(1^\lambda)$ in Section 3.4 or 3.3).

Figure 3.1: Schematic Comparison of Anonymity Experiments for Ring and Linkable Ring Signatures. (Corruption models are not specified.)

contribution is to highlight the absence of an appropriate formalism for anonymity, even in some of the most recent research.

Another model exists in the literature and has only been used for the schemes presented in Table 3.1c. This model takes better account of the anonymity expected from linkable ring signatures. It is based on an oracle and we called it anonymity (ano) as illustrated in Figure 3.1c. We recall it in Section 3.4 and show, by our upcoming counter-examples, that it is strictly stronger than 1-ano.

Linkable ring signatures admit two corruption models for linkable ring signatures:

The Honest Key model: a scenario where all signature keys must have been generated honestly by the challenger in the experiment.

The Adversarially-chosen Keys Model: a scenario where signature keys may have been generated maliciously by the adversary.

After introducing both 1-ano and ano in each of the models, we can formulate a first counter-example, showing what we claimed above: in the one-time anonymity experiment 1-ano, there are schemes revealing the identity of the signer on the second signature. We also propose a second counter-example based on existing literature [BL16]. These counter-examples are realised by proposing two constructions that could have been considered as "secure linkable ring signatures", in Section 3.5. We therefore argue for the stronger notions of anonymity ano. We discuss the insecurity of our counter-examples under this stronger model with ano-anonymity in Section 3.6. Next, we review all existing works citied in the Table 3.1a and initially based on the weaker notion of one-time anonymity 1-ano in Section 3.7. With this, we rule out a general lack of anonymity in existing constructions. By studying the proofs of existing schemes, we observed that many of them follow a similar proof pattern that can be extended by simple hybrid arguments. These results are summarised in Table 3.1.

The final contribution of this chapter is a complete classification of anonymity properties in the two corruption models for linkable ring signatures. For this, a second counter-example is needed to demonstrate the strict difference between the two corruption models. We construct it on the basis of an IND-CPA encryption scheme and any of the linkable ring signature scheme of the literature.

Related Work. Since 2004, numerous works have focused on linkable ring signature. In Table 3.1 on page 49 we provide, to the best of our knowledge, an exhaustive description of the existing linkable ring signatures in the literature, at the time of writing, while omitting signatures that have been attacked and thus provide insufficient security. These primitives claim either computational or unconditional anonymity. Most rely on discrete logarithm related assumptions, though few are based on lattice based assumptions [ATSS+18, BLO18, LNY+19, BKP20] and could achieve some post-quantum security. Some of these schemes achieve additional properties, such as *threshold* [TWC+05] or *forward-security* [BH18]. Alberto *et al.* [ATSS+18] proposed the only existing one-time linkable ring signature. However, their definition of anonymity is in fact the same as that of most linkable ring signatures. This should have given rise to concern.

All the signatures highlighted in the Table 3.1 on page 49 are based on security models adapted for individual purposes. However, these models consistently encompass a weak formalisation of the anonymity experiment, with only two works stand out with a definition that is consistent with informal descriptions [BDH+19, BKP20]. Similar realistic models have also been provided by Branco and Mateus [BM18] for *Same Ring Linkable Ring signature* and by Aranha *et al.* for *Same Message Linkable Ring Signature* [AHAN+22]. Their signatures allow more anonymity than generally considered for linkable ring signature schemes, as it limits the possibility of linking signatures in scenarios in which two signatures were generated, respectively, for the same ring or the same message. Fujisaki and Suzuki introduced a security model for *Traceable Ring signatures* [FS07a] that extends and is stronger than those considered for linkable ring signatures. Indeed, their model is similar to what was later proposed in [BDH+19] for linkable ring signatures, however, it includes additional failure conditions to prevent the adversary from trivially tracing the signer behind the challenges. All these related primitives are not strictly linkable ring signatures and their authors have not directly

provided a model adapted to linkable ring signatures. Nonetheless the general idea behind their formalism is more accurate. In order to focus only on the existing model for linkable ring signatures, we leave aside their formalism and concentrate only on the definitions that aim to formalise the security of linkable ring signatures.

Other linkable signatures have been proposed, which are based on two types of privacy preserving signatures:

Group Signatures (visually introduced in Figure 1.4 on page 15): such as *linkable group signatures* [ZLL+19], from which LRS originate, are its centralised version where an authority is responsible for managing the group. There are also weaker linkability properties, for example *selective linkability* [GL19, FGL21] which means that all signatures are unlinkable per default and only when needed, a set of signatures can be linked through the central authority. Unlike the case of ring signatures, it is possible to use hybrid arguments showing that providing one or more signatures to the adversary leads to the same property of anonymity, as the adversary has the secret signature keys of all the members of the group [BMW03]. Their decentralised equivalent also exists [FGK+22] and their anonymity is formalised in a realistic way. Diaz and Lehmann [DL21] also introduced a *user-controlled Linkable Group Signature* for which signers can provide proof of links between their signatures. With such a property, the model differs from linkable group or ring signatures as the proof of a link must be produced by the signers before a connection can be established by a verifier, and is therefore not de facto accessible. The same weakness has not passed on to their security model.

Group and Ring signatures with User-controlled Linkability: A signer of a user-controled linkable signature scheme can produce a linking witness for any of its signatures. This type of linkability was introduced by Diaz and Lehmann [DL21] for group signatures and later extended to ring signatures by Fiore *et al.* [FGK+22]. In the definitions, signatures are accompanied by a pseudonym within an event scope. Re-using the same scope leads to the same pseudonym allowing linking of signatures by the verifier. Signers can also provide explicit linking between signatures with different pseudonyms, hence allowing more linking that originally intended. In both works, the security provided by their experiment for the anonymity of the signer is analogous to the definition of anonymity ano, our arguments do not apply as they use strong anonymity notions, their schemes are not vulnerable to the exposed incorrect formulation of the anonymity experiment.

Attribute-based Signatures: *Attribute-based Signatures* [EKCGD14, EKG17] are a type of cryptographic signatures for which the signing capability is determined by the possession of certain attributes, rather than depending on the signer's public keys. This method enables the signer to demonstrate that they possess specific attributes. Attribute-based Signatures have also been proposed with user-controlled linkability. The same observation can be made as for user-controled linkable group signature. We found no weaknesses in the formalisation of anonymity in existing definitions of attribute-based signatures.

Outline. We start by presenting the most commonly used model of linkable ring signature in Section 3.2, thus formalising one-time anonymity 1-ano in both corruption models. We subsequently provide an alternative model, derived from the model of Backes *et al.* [BDH+19] for anonymity 1-ano in Section 3.3 in the *honest key* corruption model and then present the model of Backes *et al.* [BDH+19] in a stronger model in Section 3.4. In Section 3.5, we show that the models of Section 3.2 are too weak to model what is expected from a linkable ring signature. In Section 3.6, we review our counter-examples and shown them unsecure in the stronger models of Section 3.3 and 3.4. Subsequently, we review all linkable ring signature schemes to determine whether they can satisfy the stronger security requirements of Section 3.4, in Section 3.7. Last, in Section 3.8, and before concluding in Section 3.9, we provide the full relation diragram between the two anonymity properties (1-ano and ano) in the two corruption models.

3.2 Review of Linkable Ring Signatures Definitions

Definitions of *linkable ring signatures* vary across the literature (see references given in Table 3.1 on page 49). Despite that, the prescribed algorithms have been defined in the same way in almost all presented works. This is not always the case for their associated security definitions, even if they remain relatively similar.

Definition 23: Linkable Ring Signature - LRS

A *Linkable Ring Signature* scheme is composed of five algorithms defined as follows:

$\mathsf{Setup}_{\mathsf{LRS}}(1^\lambda)$**:** is a PPT algorithm that takes the security parameter λ and produces the *public parameters* pp.

We assume these parameters pp as common inputs to all the upcoming algorithms.

$\mathsf{Gen}_{\mathsf{LRS}}(1^\lambda)$**:** is a PPT algorithm that takes the security parameter λ, and it returns a pair of keys $(\mathsf{pk}, \mathsf{sk})$.

$\mathsf{Sign}_{\mathsf{LRS}}(\mathsf{sk}_i, m, \{\mathsf{pk}_j\}_{j\in\mathcal{R}})$**:** is a PPT algorithm that takes a *public key set* $\{\mathsf{pk}_i\}_{i\in\mathcal{R}}$ for a ring set $\mathcal{R}$, a signer secret key sk_i (with $i \in \mathcal{R}$) and a message m. It returns a ring signature σ.

$\mathsf{Verif}_{\mathsf{LRS}}(m, \sigma, \{\mathsf{pk}_i\}_{i\in\mathcal{R}})$**:** is a deterministic polynomial-time algorithm that takes a public key set $\{\mathsf{pk}_i\}_{i\in\mathcal{R}}$, a signature σ, and a message m. If the signature σ is valid, then it returns 1, otherwise, it returns 0.

$\mathsf{Link}_{\mathsf{LRS}}(\sigma, \sigma')$**:** is a deterministic polynomial-time algorithm that takes two signatures σ and σ', it returns 1 if they are linked, otherwise, it returns 0.

A linkable ring signature must guarantee *Correctness*, *Unforgeability*, *One-time Anonymity*, *Linkability* and *Non-slanderability* as defined below.

Correctness. Honestly generated signatures on any message m should verify the equation:

$$\forall \lambda, \forall \mathcal{R} \subset \mathbb{N}, \forall i \in \mathcal{R}, \forall \mathsf{pp} \in [\mathsf{Setup}_{\mathsf{LRS}}(1^\lambda)], \forall (\mathsf{pk}_j, \mathsf{sk}_j)_{j \in \mathcal{R}} \in [\mathsf{Gen}_{\mathsf{LRS}}(1^\lambda)]^{|\mathcal{R}|},$$
$$\forall \sigma \in [\mathsf{Sign}_{\mathsf{LRS}}(\mathsf{sk}_i, m, \{\mathsf{pk}_j\}_{j \in \mathcal{R}})], \mathsf{Verif}_{\mathsf{LRS}}(m, \sigma, \{\mathsf{pk}_j\}_{j \in \mathcal{R}}) = 1.$$

As discussed in depth in [BKM06], the corruption model of RS, in particular the anonymity of the signer, can be based on different corruption setups, from the weakest to the strongest:

Honest Key Model (HK). The *Honest Key Model* assumes that all the keys within the rings are generated honestly by the challenger. They may later be corrupted by the adversary. Consequently, no security is provided against keys generated maliciously.

Adversarially-Chosen Keys Model (ACK). The *Adversarially-Chosen Keys Model* allows the adversary to supply maliciously generated keys to the signing oracle and the challenge signature ring, hence dropping the assumption that all keys need to be generated honestly. This model solves the problem of the honest key model by assuming that keys could have been generated maliciously by the signers. However, it does not guarantee that the entities in the ring are unable to identify the signer if they all collude, including the signer, *i.e.,* if all secret keys are revealed to the adversary.

Full Key Exposure Model (FKE). The *Full Key Exposure Model* was proposed for ring signatures, assuming full disclosure of all secret keys to the adversary. In the context of ring signatures this model ensured anonymity even in case of leakage of all the secret keys. However, this level of security cannot be achieved for linkable ring signature schemes: given knowledge of all the secret key, the adversary can generate signatures with every single keys and use the $\mathsf{Link}_{\mathsf{LRS}}$ algorithm to identify the signers.

These corruption models, originally proposed for ring signatures, also apply to linkable ring signatures, with the exception of the full key exposure model. We elucidate on this fact at the end of this section, after presenting the property of anonymity.

Like in all the previous models proposed by the papers listed in Table 3.1a, the security of LRS is introduced here in the *honest key model*, *i.e.,* all keys must have been generated honestly by the challenger and only some of them can be corrupted based on a corruption oracle provided to the adversary. The honest key model leads to a weak corruption model, contradicting the *ad hoc* purpose of ring signatures, as any signer may generate its own key without any checks by other parties. We first introduce the definition of the required oracles before presenting the four game-based security requirements for *Secure Linkable Ring Signatures*. We discuss the model provided by Backes *et al.* [BDH$^+$19] in Sections3.4 and 3.3 and also discuss the more apropriate adversarially-chosen keys model.

Oracles. The adversary has access to the following oracles when it attempts to break the security of a linkable ring signature scheme.

$\mathcal{JO}$**.** The *Joining Oracle*. Given the security parameter λ, runs $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}_{\mathsf{LRS}}(1^\lambda)$ and outputs the public key pk.

$\mathcal{CO}$**.** The *Corruption Oracle*. Given a public key pk which is the output of a previous query to $\mathcal{JO}$, $\mathcal{CO}$ returns its corresponding secret key sk.

$\mathcal{SO}$**.** The *Signature Oracle*. Given a public key vector $\{\mathsf{pk}_i\}_{i\in\mathcal{R}}$ an insider public key pk_i, for $i \in \mathcal{R}$ previously generated by $\mathcal{JO}$, and a message m, $\mathcal{SO}$ returns the signature $\sigma \leftarrow \mathsf{Sign}_{\mathsf{LRS}}(\mathsf{sk}_i, m, \{\mathsf{pk}_i\}_{i\in\mathcal{R}})$ and keeps record of the signed messages m in the set $\mathcal{SO}$.

For notation purposes, in our security experiments, we use the above oracle to designate the set of public keys of the entity that have been either introduced into the oracle or generated by it for $\mathcal{JO}$. The set $\mathcal{SO}$ records multiple types of elements:

Messages: $\mathcal{SO}$ records the set of messages input to the oracle, when we write $m \in \mathcal{SO}$ for m a message, or;

Messages and signatures: $\mathcal{SO}$ records the set of message-signature pairs input to the oracle, when we write $(m, \sigma) \in \mathcal{SO}$ for m a message and σ the associated signature, or;

Public keys of signers: $\mathcal{SO}$ records the set of public keys of the signers which produced the linkable ring signatures when the oracle is called, when we write $\mathsf{pk} \in \mathcal{SO}$ for the public key pk of a signer.

Security Model. We now describe the properties expected for linkable ring signatures, namely *unforgeability*, *one-time anonymity*, *linkability* and *non-slanderability*. We always denote by $\mathsf{Adv}^{\mathsf{prop}}_{\mathcal{A},\mathsf{LRS}}(1^\lambda)$ the advantage of $\mathcal{A}$ against the property prop of a linkable ring signature LRS for a given security parameter λ. Experiences are provided in the honest key model.

Unforgeability (unf-HK). Constructing a valid signature without using the secret key should be unfeasible. Formally, the probability $\mathsf{Adv}^{\mathsf{unf\text{-}HK}}_{\mathcal{A},\mathsf{LRS}}(1^\lambda)$ of a PPT adversary $\mathcal{A}$ winning (*i.e.,* making the challenger return 1) against the experiment $\mathsf{Exp}^{\mathsf{unf\text{-}HK}}_{\mathcal{A},\mathsf{LRS}}(1^\lambda)$ should be negligible in the security parameter λ. Note that we could instead require a stronger variant, where a new signature on a signed messages would be accepted as a forgery. For that, a record of the messages input to the signature oracle and the output signature is kept in the set $\mathcal{SO}$, and line 5 of $\mathsf{Exp}^{\mathsf{unf\text{-}HK}}_{\mathcal{A},\mathsf{LRS}}(1^\lambda)$ checks if $(m^*, \sigma^*) \in \mathcal{SO}$ instead.

$\mathsf{Exp}^{\mathsf{unf\text{-}HK}}_{\mathcal{A},\mathsf{LRS}}(1^\lambda)$ - (Unforgeability Experiment in the Honest Key Model)

1 : $\mathsf{pp} \leftarrow \mathsf{Setup}_{\mathsf{LRS}}(1^\lambda)$
2 : $(m^*, \sigma^*, (\mathsf{pk}_i)_{i\in\mathcal{R}}) \leftarrow \mathcal{A}^{\mathcal{JO},\mathcal{CO},\mathcal{SO}}(\mathsf{pp})$
3 : **if** $\{\mathsf{pk}_i\}_{i\in\mathcal{R}} \not\subset \mathcal{JO}$: **return** 0
// All of the public keys in $\{\mathsf{pk}_i\}_{i\in\mathcal{R}}$ were output by $\mathcal{JO}$.
4 : **if** $\{\mathsf{pk}_i\}_{i\in\mathcal{R}} \cap \mathcal{CO} \neq \emptyset$: **return** 0 // No public key in $(\mathsf{pk}_i)_{i\in\mathcal{R}}$ were queried to $\mathcal{CO}$.
5 : **if** $m^* \in \mathcal{SO}$: **return** 0 // The message m^* was not an input to $\mathcal{SO}$.
6 : **return** $\mathsf{Verif}_{\mathsf{LRS}}(m^*, \sigma^*, \{\mathsf{pk}_i\}_{i\in\mathcal{R}}) = 1$

One-time Anonymity (1-ano -HK) *(previously named anonymity)*. It must be difficult to guess the public key corresponding to the secret key used to produce a signer's first signature. Here we present the property generally provided in the literature and call it *One-time Anonymity* whereas the property was previously given as *Anonymity*. Formally, for any PPT adversary $\mathcal{A}$, the experiment $\mathsf{Exp}^{\mathsf{1\text{-}ano\text{-}HK}}_{\mathcal{A},\mathsf{LRS}}(1^\lambda)$ should have a negligible probability to output 1:

$$\mathsf{Adv}^{\mathsf{1\text{-}ano\text{-}HK}}_{\mathcal{A},\mathsf{LRS}}(1^\lambda) = |\Pr[\mathsf{Exp}^{\mathsf{1\text{-}ano\text{-}HK}}_{\mathcal{A},\mathsf{LRS}}(1^\lambda) = 1] - 1/2| \leq \epsilon(1^\lambda).$$

$\mathsf{Exp}^{\mathsf{1\text{-}ano\text{-}HK}}_{\mathcal{A},\mathsf{LRS}}(1^\lambda)$ - (One-time Anonymity Experiment in the Honest Key Model)

1 : $\mathsf{pp} \leftarrow \mathsf{Setup}_{\mathsf{LRS}}(1^\lambda)$
2 : $(m^*, (\mathsf{pk}_i)_{i\in\mathcal{R}^*}, i_0, i_1) \leftarrow \mathcal{A}^{\mathcal{JO},\mathcal{CO},\mathcal{SO}}(\mathsf{pp})$
3 : $b \xleftarrow{\$} \{0,1\}^*$
4 : $\sigma \leftarrow \mathsf{Sign}_{\mathsf{LRS}}(\mathsf{sk}_{i_b}, m^*, \{\mathsf{pk}_i\}_{i\in\mathcal{R}^*} \cup \{\mathsf{pk}_{i_0}, \mathsf{pk}_{i_1}\})$
5 : $b^* \leftarrow \mathcal{A}^{\mathcal{JO},\mathcal{CO},\mathcal{SO}}(\sigma)$
6 : **if** $\{\mathsf{pk}_i\}_{i\in\mathcal{R}^*} \not\subset \mathcal{JO}$: **return** b // All of the public keys in $(\mathsf{pk}_i)_{i\in\mathcal{R}^*}$ are outputs of $\mathcal{JO}$.
7 : **if** $\{\mathsf{pk}_i\}_{i\in\mathcal{R}^*} \cap \mathcal{CO} \neq \emptyset$: **return** b // No public key in $(\mathsf{pk}_i)_{i\in\mathcal{R}^*}$ was queried to $\mathcal{CO}$.
8 : **if** $\{\mathsf{pk}_{i_0}, \mathsf{pk}_{i_1}\} \cap \mathcal{SO} \neq \emptyset$: **return** b // The oracle $\mathcal{SO}$ did not allow the link.
9 : **return** $b = b^*$

This property only allows the adversary $\mathcal{A}$ to obtain a single signature σ produced by the signer associated with the key pk_{i_b} and no signature from the signer associated with the key $\mathsf{pk}_{i_{(1-b)}}$ (line 4 and 5 of $\mathsf{Exp}^{\mathsf{1\text{-}ano\text{-}HK}}_{\mathcal{A},\mathsf{LRS}}(1^\lambda)$). Consequently, the property offers no guarantees on the anonymity of the signer when several signatures are produced with the same keys. This formalism contradicts the intended use for LRS, which is designed for different use cases than one-time LRS. In particular, the anonymity of the signer is expected to persist throughout the lifespan of the keys.

Some models, such as in [LASZ13], are even weaker and assume that none of the members of the challenge ring (*i.e.,* all entities associated with keys in $\{\mathsf{pk}_i\}_{i\in\mathcal{R}^*}$) have ever produced a signature with their keys. These definitions do not reflect the actual use of linkable ring signatures, as this primitive was designed to allow multiple anonymous signatures for a single entity. In Section 3.5, we give further arguments and two counter-examples for obtaining the above property, but without what was informally described

as *anonymity* (see Section 3.1). This shows the limits of the experiment proposed above as $\mathsf{Exp}^{\text{1-ano-HK}}_{\mathcal{A},\mathsf{LRS}}(1^\lambda)$.

Linkability (link -HK). It must be difficult to generate two unlinked valid signatures from the same signer. To obtain linkability, the probability $\mathsf{Adv}^{\text{link-HK}}_{\mathcal{A},\mathsf{LRS}}(1^\lambda)$ of winning the experiment $\mathsf{Exp}^{\text{link-HK}}_{\mathcal{A},\mathsf{LRS}}(1^\lambda)$ must be negligible.

$\mathsf{Exp}^{\text{link-HK}}_{\mathcal{A},\mathsf{LRS}}(1^\lambda)$ - (Linkability Experiment in the Honest Key Model)

1 : $\mathsf{pp} \leftarrow \mathsf{Setup}_{\mathsf{LRS}}(1^\lambda)$

2 : $(m_0^*, \sigma_0^*, (\mathsf{pk}_i)_{i\in\mathcal{R}_0^*}), (m_1^*, \sigma_1^*, (\mathsf{pk}_i)_{i\in\mathcal{R}_1^*}) \leftarrow \mathcal{A}^{\mathcal{JO},\mathcal{CO},\mathcal{SO}}(\mathsf{pp})$

3 : **if** $\{\mathsf{pk}_i\}_{i\in\mathcal{R}_0^*\cup\mathcal{R}_1^*} \not\subset \mathcal{JO}$: **return** 0

// Public keys in $(\mathsf{pk}_i)_{i\in\mathcal{R}_0^*\cup\mathcal{R}_1^*}$ are honestly generated.

4 : **if** $\exists i,j \in \mathcal{R}_0^* \cup \mathcal{R}_1^*, i \neq j, \mathsf{pk}_i, \mathsf{pk}_j \in \mathcal{CO}$: **return** 0

// Max. one corrupted key in the rings.

5 : **if** $\{\mathsf{pk}_i\}_{\mathcal{R}_0^*\cup\mathcal{R}_1^*} \cap \mathcal{SO} \neq \emptyset$: **return** 0

// The oracle $\mathcal{SO}$ did not return a linked signature.

6 : **return** $\mathsf{Verif}_{\mathsf{LRS}}(m_0^*, \sigma_0^*, \{\mathsf{pk}_i\}_{i\in\mathcal{R}_0^*}) = \mathsf{Verif}_{\mathsf{LRS}}(m_1^*, \sigma_1^*, \{\mathsf{pk}_i\}_{i\in\mathcal{R}_1^*}) = 1$
$\wedge\, \mathsf{Link}_{\mathsf{LRS}}(\sigma_0^*, \sigma_1^*) = 0$

Non-slanderability (slan-HK). It should be unfeasible to link two valid signatures correctly generated by different signers. To obtain non-slanderability, the probability $\mathsf{Adv}^{\text{slan-HK}}_{\mathcal{A},\mathsf{LRS}}(1^\lambda)$ of winning the experiment $\mathsf{Exp}^{\text{slan-HK}}_{\mathcal{A},\mathsf{LRS}}(1^\lambda)$ must be negligible.

$\mathsf{Exp}^{\text{slan-HK}}_{\mathcal{A},\mathsf{LRS}}(1^\lambda)$ - (Non-slanderability Experiment in the Honest Key Model)

1 : $\mathsf{pp} \leftarrow \mathsf{Setup}_{\mathsf{LRS}}(1^\lambda)$

2 : $(\mathsf{pk}^*, m_0^*, \{\mathsf{pk}_i\}_{i\in\mathcal{R}_0^*}) \leftarrow \mathcal{A}^{\mathcal{JO},\mathcal{CO},\mathcal{SO}}(\mathsf{pp})$

3 : **if** $\{\mathsf{pk}^*\} \cup \{\mathsf{pk}_i\}_{i\in\mathcal{R}_0^*} \not\subset \mathcal{JO}$: **return** 0

// The key pk^* is honestly generated and the set $\mathcal{R}_0^*$ only contains honestly generated keys.

4 : $\sigma \leftarrow \mathsf{Sign}_{\mathsf{LRS}}(\mathsf{sk}, m_0^*, \{\mathsf{pk}^*\} \cup \{\mathsf{pk}_i\}_{i\in\mathcal{R}_0^*})$ // sk is the secret key associated to pk.

5 : $(m_1^*, \sigma^*, \{\mathsf{pk}_i\}_{i\in\mathcal{R}_1^*}) \leftarrow \mathcal{A}^{\mathcal{JO},\mathcal{CO},\mathcal{SO}}(\sigma)$

6 : **if** $\{\mathsf{pk}_i\}_{i\in\mathcal{R}_1^*} \not\subset \mathcal{JO}$: **return** 0 // The set $\mathcal{R}_1^*$ only contains honestly generated keys.

7 : **if** $\mathsf{pk}^* \in \mathcal{CO}$: **return** 0 // The public key pk^* has not been requested from $\mathcal{CO}$.

8 : **if** $\mathsf{pk}^* \in \mathcal{SO}$: **return** 0 // The oracle $\mathcal{SO}$ did not produce the signature σ^*.

9 : **return** $\mathsf{Verif}_{\mathsf{LRS}}(m_1^*, \sigma^*, \{\mathsf{pk}_i\}_{i\in\mathcal{R}_1^*}) = 1 \wedge \mathsf{Link}_{\mathsf{LRS}}(\sigma, \sigma^*) = 1$

Some definitions of non-slanderability, such as the one in [ATSS+18] require $\mathcal{A}$ to use specific keys to generate a signature. We deviate slightly from that definition by prohibiting corruption of the entity targeted by the attack, but follows the main idea of that formalisation.

From here on we can note that the correctness of the linking algorithm $\mathsf{Link}_{\mathsf{LRS}}$ is guaranteed by the properties of *linkability* and *non-slanderability*.

Unconditional Variant. We say that a property prop is obtained unconditionally if, for any unbounded probabilistic adversary $\mathcal{A}$, its advantage $\mathsf{Adv}^{\mathsf{prop}}_{\mathcal{A},\mathsf{LRS}}(1^\lambda)$ is equal to 0.

Amongst existing work (see Table 3.1 on page 49), only a few schemes, such as [LASZ13, BH18, ATSS+18, BBG+22], have achieved unconditional one-time anonymity 1-ano.

Adversarially-chosen Keys Model. As stated above, most existing work listed in Table 3.1a sets unusually low security requirements. All of the security experiments presented in this section and in all previous work refered to in Table 3.1a are modelled within the framework of the honest key model HK, hence, failing to take into consideration the possibility of potentially malicious *adversarially generated keys*, the ACK model. This is inconsistent with informal security expectations for LRS as already stated. We present the experiment for one-time anonymity in the 1-ano in the ACK corruption model below.

$\mathsf{Exp}^{\text{1-ano-ACK}}_{\mathcal{A},\mathsf{LRS}}(1^\lambda, n)$ -
(One-time Anonymity experiment with respect to adversarially-chosen keys)

$\mathsf{pp} \leftarrow \mathsf{Setup}_{\mathsf{LRS}}(1^\lambda)$

$\{\mathsf{pk}_i, \mathsf{sk}_i\}_{i=1}^n \leftarrow \mathsf{Gen}_{\mathsf{LRS}}(1^\lambda)$ // Abusing notations, the algorithm is executed n times.

$(m^*, (\mathsf{pk}_i)_{i\in\mathcal{R}^*}, i_0, i_1) \leftarrow \mathcal{A}^{\mathcal{SO}}(\mathsf{pp}, \{\mathsf{pk}_i\}_{i=1}^n)$

// The set $\mathcal{R}$ for which $\mathcal{SO}$ is queried can also contain public keys picked by the adversary.

$b \xleftarrow{\$} \{0,1\}$

$\sigma \leftarrow \mathsf{Sign}_{\mathsf{LRS}}(\mathsf{sk}_{i_b}, m^*, \{\mathsf{pk}_i\}_{i\in\mathcal{R}^*} \cup \{\mathsf{pk}_0, \mathsf{pk}_1\})$

$b^* \leftarrow \mathcal{A}^{\mathcal{SO}}(\sigma)$

if $\{\mathsf{pk}_{i_0}, \mathsf{pk}_{i_1}\} \cap \mathcal{SO} \neq \emptyset$: **return** b // The oracle $\mathcal{SO}$ did not allow any link.

return $b = b^*$

The introduction of the other properties, unf-ACK, link-ACK, slan-ACK is postponed to Figure 3.2 on page 62 in Section 3.4 with the introduction of Backes *et al.* [BDH+19]'s property of anonymity ano.

The full key exposure corruption model, which is stronger than the adversarially-chosen keys corruption model, cannot be achieved for linkable ring signatures. This is because anonymity of LRS cannot be achieved in the full key exposure corruption model. Linkable ring signature are always claimable, *e.g.,* by performing a signature for any given message and using the link algorithm anyone can test if they were both produced by the same signer. Therefore, revealing the challenger's secret key always breaks anonymity.

3.3 Anonymity in the Honest-Key Model

The security properties of ring signatures were formalised in a work by Bender *et al.* [BKM06]. In particular, unforgeability and anonymity of ring signatures were extensively studied in this research paper. Their models encompass three levels of corruptions. The honest key model is the most considered one for linkable ring signature and always with the flawed one-time anonymity experiments. Only two works [BDH+19, BKP20] stand out and consider linkable ring signatures in the adversarially-chosen keys model that we will introduce later in Section 3.4. Moreover, their definition of anonymity, that of the second [BKP20] resulting from the first [BDH+19], is the only one in the literature

to consider a natural and stronger formalisation of anonymity for linkable ring signatures. They take advantage of what is sometimes called a *Left-or-Right* ($\mathcal{LoR}$) oracle. It acts as a challenge oracle providing signatures to the adversary for consistent unknown *left* and *right* signers. The adversary must uncover how the identity of the two signers are distributed in between the two challenger signers. The $\mathcal{LoR}^{\mathsf{HK}}$ oracles is defined in a context in which two key pairs $(\mathsf{pk}_{i_0}, \mathsf{sk}_{i_0})$ and $(\mathsf{pk}_{i_1}, \mathsf{sk}_{i_1})$ are known by the challenger, which also holds a bit $b \in \{0, 1\}$. The oracle is defined as follows:

$\mathcal{LoR}^{\mathsf{HK}}$**.** The *Left-or-Right* oracle $\mathcal{LoR}_b^{\mathsf{HK}}(\cdot, \cdot)$ is such that for a call $\mathcal{LoR}_b^{\mathsf{HK}}(m, \{\mathsf{pk}_i\}_{i\in\mathcal{R}})$, it checks that all the public keys $\{\mathsf{pk}_i\}_{i\in\mathcal{R}}$ were honestly generated, hence belongs to $\mathcal{JO}$, and if so, it returns a signature $\mathsf{Sign}_{\mathsf{LRS}}(\mathsf{sk}_{i_b}, m, \{\mathsf{pk}_i\}_{i\in\mathcal{R}} \cup \{\mathsf{pk}_{i_0}, \mathsf{pk}_{i_1}\})$.

The $\mathcal{LoR}^{\mathsf{HK}}$ oracle can be queried for any arbitrary set of registered keys $\{pk_i\}_{i\in\mathcal{R}}$. This set is always supplemented by the key of the two challengers, pk_{i_0} and pk_{i_1}, in order to avoid trivial identification attacks based on the failure of the oracles.

We introduce the definition of anonymity for linkable ring signatures as per [BDH+19] in the *honest-key model*. For the anonymity under the honest key model to hold against a PPT adversary $\mathcal{A}$, it should be computationally difficult to guess the public key corresponding to the secret key used during the production of the signatures of a signer. Formally, the experiment $\mathsf{Exp}^{\mathsf{ano}}_{\mathcal{A},\mathsf{LRS}}(1^\lambda, n)$ should have a negligible probability $\mathsf{Adv}^{\mathsf{ano}}_{\mathcal{A},\mathsf{LRS}}(1^\lambda, n)$ given by:

$$\mathsf{Adv}^{\mathsf{ano}}_{\mathcal{A},\mathsf{LRS}}(1^\lambda, n) = |\Pr[\mathsf{Exp}^{\mathsf{ano\text{-}HK}}_{\mathcal{A},\mathsf{LRS}}(1^\lambda, n) = 1] - 1/2| \leq \epsilon(1^\lambda).$$

This bound must hold for every $n \in \mathbb{N}$ and the following experiment.

$\mathsf{Exp}^{\mathsf{ano\text{-}HK}}_{\mathcal{A},\mathsf{LRS}}(1^\lambda, n)$ - (Anonymity in the honest keys model)

$\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$

$\{\mathsf{pk}_i, \mathsf{sk}_i\}_{i=1}^n \leftarrow \mathsf{Gen}_{\mathsf{LRS}}(1^\lambda)$

$(m^*, i_0, i_1) \leftarrow \mathcal{A}^{\mathcal{SO}}(\mathsf{pp}, \{\mathsf{pk}_i\}_{i=1}^n)$ // Requests $\mathcal{SO}$ must be made using the provided keys.

$b \xleftarrow{\$} \{0, 1\}$

$b^* \leftarrow \mathcal{A}^{\mathcal{SO}, \mathcal{LoR}_b^{\mathsf{HK}}}(1^\lambda)$

if $\{\mathsf{pk}_{i_0}, \mathsf{pk}_{i_1}\} \cap \mathcal{SO} \neq \emptyset$: **return** b

// The $\mathcal{SO}$ oracle did not output a signature for the signer pk_{i_b}.

if $\mathcal{SO}$ was queried for a ring $\mathcal{R}$ with a public key which is not in $\{\mathsf{pk}_i\}_{i=1}^n$:

  return b

if $\mathcal{LoR}^{\mathsf{HK}}$ was queried for a ring $\mathcal{R}$ with a public key which is not in $\{\mathsf{pk}_i\}_{i=1}^n$:

  return b

return $b = b^*$

In this experiment, the challenge is not directly sent to the adversary, but is deported to the answers of the $\mathcal{LoR}^{\mathsf{HK}}$ oracle which provides challenges as output when called by the adversary. Therefore, when proving the anonymity of LRS under this model, every execution of the oracle $\mathcal{LoR}^{\mathsf{HK}}$ would have to be considered by the reduction instead of just the first signature, which could lead to less tight reductions when these reductions

are not unconditional. However, it does more accurately formalise the anonymity of the linkable ring signature than has previously been achieved in the literature.

Definition 24: Linkable Ring Signature in the Honest-key Model

A *Linkable Ring Signature* scheme is defined with algorithms described in Definition 23 on page 54 and achieves security in the *honest-key model* if it achieves the properties of *Unforgeability* unf-HK, *Linkability* link-HK and *Non-slanderability* slan-HK as described in Section 3.2 and *Anonymity* ano-HK as described above in this Section.

This model with anonymity formalised in the honest key model can only be used when key generation is fully trusted. The use cases are then either (1) when it is possible to prove the honesty of the key generations, or (2) when all the members of the ring are honest. While this assumption may be realistic for some threat models, ring signatures are, by their nature, intended for use in contexts where there is no central authority responsible for verifying the validity of public keys, otherwise linkable group signatures could be used [ZLL+19]. As a result, this definition does not always reflect the actual security requirements for linkable ring signatures, especially when used in decentralised scenarios such as blockchains [ATSS+18]. This model leaves open possible attack scenarios in which (1) an adversary arbitrarily generates public keys (which may possibly depend on the public keys of honest users), and then (2) a legitimate signer generates a signature for a ring containing some of these adversary-generated public keys. Definition 24 offers no protection in these scenarios. This motivates the use of a stronger definition in the adversary-selected key model.

3.4 Anonymity of Linkable Ring Signatures

Technical Summary

Despite more than 20 years of research in this area, misconceptions have persisted about the one-time anonymity experiment for linkable ring signatures. At the time of writing, only two works [BDH+19, BKP20] have considered realistic models: in the adversarially-chosen keys model ACK and with anonymity formalised based on a *Left-or-Right* ($\mathcal{LoR}$) signer oracle as a challenge. This oracle allows to provide multiple signatures from the challenger to the adversary. This accurate model has largely been overlooked in subsequent work, despite seemingly being achieved by most linkable ring signatures. We restate their definition, demonstrating its precision and that the introduction of the $\mathcal{LoR}$ oracle excludes the counter-examples later presented in Section 3.5 and demonstrated our claim of weakness of the definition of one-time anonymity 1-ano.

For the formalisation of the security properties of linkable ring signatures in the adversarially-chosen keys model ACK, we instantiate two oracles: the $\mathcal{SO}^{\mathsf{ACK}}$ and the $\mathcal{LoR}^{\mathsf{ACK}}$ oracles. They are both defined below. The $\mathcal{LoR}^{\mathsf{ACK}}$ oracles is defined in a

$\mathsf{Exp}^{\mathsf{ano\text{-}ACK}}_{\mathcal{A},\mathsf{LRS}}(1^\lambda, n)$ - (Anonymity Experiment in the Adversarially-chosen Keys Model)

$\mathsf{pp} \leftarrow \mathsf{Setup}_{\mathsf{LRS}}(1^\lambda)$

$\{\mathsf{pk}_i, \mathsf{sk}_i\}_{i=1}^n \leftarrow \mathsf{Gen}_{\mathsf{LRS}}(1^\lambda)$

$(i_0, i_1) \leftarrow \mathcal{A}^{\mathcal{SO}^{\mathsf{ACK}}}(\mathsf{pp}, \{\mathsf{pk}_i\}_{i=1}^n)$

// The set $\mathcal{R}$ for which $\mathcal{SO}^{\mathsf{ACK}}$ is queried can also contain public keys picked by the adversary.

$b \xleftarrow{\$} \{0,1\}$

$b^* \leftarrow \mathcal{A}^{\mathcal{SO}^{\mathsf{ACK}}, \mathcal{LO}\mathcal{R}_b^{\mathsf{ACK}}}(1^\lambda)$

if $\{\mathsf{pk}_{i_0}, \mathsf{pk}_{i_1}\} \cap \mathcal{SO}^{\mathsf{ACK}} \neq \emptyset$: **return** b // The oracle $\mathcal{SO}^{\mathsf{ACK}}$ did not allow any link.

return $b = b^*$

$\mathsf{Exp}^{\mathsf{unf\text{-}ACK}}_{\mathcal{A},\mathsf{LRS}}(1^\lambda)$ - (Unforgeability Experiment in the Adversarially-chosen Keys Model)

$\mathsf{pp} \leftarrow \mathsf{Setup}_{\mathsf{LRS}}(1^\lambda)$

$\{\mathsf{pk}_i, \mathsf{sk}_i\}_{i=1}^n \leftarrow \mathsf{Gen}_{\mathsf{LRS}}(1^\lambda)$

$(m^*, \sigma^*, \{\mathsf{pk}_i\}_{i\in\mathcal{R}}) \leftarrow \mathcal{A}^{\mathcal{SO}^{\mathsf{ACK}}}(\mathsf{pp}, (\mathsf{pk}_i)_{1\leq i\leq n})$

if $\mathcal{R} \not\subset \{1, \ldots, n\}$: **return** 0 // No corrupted public key in the ring.

if $m^* \in \mathcal{SO}^{\mathsf{ACK}}$: **return** 0 // The message m^* has not been an input of $\mathcal{SO}^{\mathsf{ACK}}$.

return $\mathsf{Verif}_{\mathsf{LRS}}(m^*, \sigma^*, \{\mathsf{pk}_i\}_{i\in\mathcal{R}}) = 1$

$\mathsf{Exp}^{\mathsf{link\text{-}ACK}}_{\mathcal{A},\mathsf{LRS}}(1^\lambda, n)$ - (Linkability Experiment in the Adversarially-chosen Keys Model)

$\mathsf{pp} \leftarrow \mathsf{Setup}_{\mathsf{LRS}}(1^\lambda)$

$\{\mathsf{pk}_i, \mathsf{sk}_i\}_{i=1}^n \leftarrow \mathsf{Gen}_{\mathsf{LRS}}(1^\lambda)$

$(m_0^*, \sigma_0^*, \{\mathsf{pk}_i^*\}_{i\in\mathcal{R}_0^*}), (m_1^*, \sigma_1^*, \{\mathsf{pk}_i^*\}_{i\in\mathcal{R}_1^*}) \leftarrow \mathcal{A}^{\mathcal{SO}^{\mathsf{ACK}}}(\mathsf{pp}, \{\mathsf{pk}_i\}_{1\leq i\leq n})$

if $\exists i \in \mathcal{R}_0^*, \exists j \in \mathcal{R}_1^*, \mathsf{pk}_i \neq \mathsf{pk}_j^*, \mathsf{pk}_i^*, \mathsf{pk}_j^* \notin \{\mathsf{pk}_i\}_{1\leq i\leq n}$: **return** 0

// Only one common corrupted key or many in the same ring.

if (m_0, σ_0^*) or $(m_1, \sigma_1^*) \in \mathcal{SO}^{\mathsf{ACK}}$: **return** 0

// The oracle $\mathcal{SO}^{\mathsf{ACK}}$ did not produce the signatures.

return $\mathsf{Verif}_{\mathsf{LRS}}(m_0^*, \sigma_0^*, \{\mathsf{pk}_i^*\}_{i\in\mathcal{R}_0^*}) = 1 \wedge \mathsf{Verif}_{\mathsf{LRS}}(m_1^*, \sigma_1^*, \{\mathsf{pk}_i^*\}_{i\in\mathcal{R}_1^*}) = 1$
$\wedge\, \mathsf{Link}_{\mathsf{LRS}}(\sigma_0^*, \sigma_1^*) = 0$

$\mathsf{Exp}^{\mathsf{slan\text{-}ACK}}_{\mathcal{A},\mathsf{LRS}}(1^\lambda, n)$ - (Non-slanderability Experiment in the Adversarially-chosen Keys Model)

$\mathsf{pp} \leftarrow \mathsf{Setup}_{\mathsf{LRS}}(1^\lambda)$

$\{\mathsf{pk}_i, \mathsf{sk}_i\}_{i=1}^n \leftarrow \mathsf{Gen}_{\mathsf{LRS}}(1^\lambda)$

$i^*, m_0^*, \{\mathsf{pk}_i^*\}_{\mathcal{R}_0^*} \leftarrow \mathcal{A}^{\mathcal{SO}^{\mathsf{ACK}}}(\mathsf{pp}, \{\mathsf{pk}_k\}_{1\leq k\leq n})$

if $i^* \notin \{1, \ldots, n\}$: **return** 0 // The designated signer has been produced by the challenger.

$\sigma \leftarrow \mathsf{Sign}_{\mathsf{LRS}}(\mathsf{sk}_{i^*}, m_0^*, \{\mathsf{pk}_i^*\}_{i\in\mathcal{R}_0^*})$

$m_1^*, \sigma^*, \mathcal{R}_1^* \leftarrow \mathcal{A}^{\mathcal{SO}^{\mathsf{ACK}}}(\sigma)$

if $\mathsf{pk}_{i^*} \in \mathcal{SO}^{\mathsf{ACK}}$: **return** 0

// The oracle $\mathcal{SO}^{\mathsf{ACK}}$ did not allowed to produce the signature σ^* for the key pk_{i^*}.

return $\mathsf{Verif}_{\mathsf{LRS}}(m_1^*, \sigma^*, \{\mathsf{pk}_i^*\}_{i\in\mathcal{R}_1^*}) = 1 \wedge \mathsf{Link}_{\mathsf{LRS}}(\sigma, \sigma^*) = 1$

Figure 3.2: Experiments for Anonymity, Unforgeability, Linlability and Non-slanderability in the Adversarially-chosen Keys Model. (Similar to the one given in [BDH+19].)

context where two key pairs $(\mathsf{pk}_{i_0}, \mathsf{sk}_{i_0})$ and $(\mathsf{pk}_{i_1}, \mathsf{sk}_{i_1})$ are known by the challenger as well as a bit $b \in \{0, 1\}$.

$\mathcal{SO}^{\mathsf{ACK}}$. The oracle $\mathcal{SO}^{\mathsf{ACK}}(\cdot, \cdot, \cdot)$ is such that for a call $\mathcal{SO}^{\mathsf{ACK}}(i, m, \mathcal{R})$, it returns $\mathsf{Sign}_{\mathsf{LRS}}(\mathsf{sk}_i, m, \{\mathsf{pk}_i\}_{i \in \mathcal{R}})$, where sk_i must be known by the challenger and $i \in \mathcal{R}$.

$\mathcal{LOR}^{\mathsf{ACK}}$. For two honestly generated key pairs $(\mathsf{pk}_{i_0}, \mathsf{sk}_{i_0})$ and $(\mathsf{pk}_{i_1}, \mathsf{sk}_{i_1})$. The *Left-or-Right* oracle $\mathcal{LOR}_b^{\mathsf{ACK}}(\cdot, \cdot)$ is such that for a call $\mathcal{LOR}_b^{\mathsf{ACK}}(m, \{\mathsf{pk}_i\}_{i \in \mathcal{R}})$, it returns a signature $\mathsf{Sign}_{\mathsf{LRS}}(\mathsf{sk}_{i_b}, m, \{\mathsf{pk}_i\}_{i \in \mathcal{R}} \cup \{\mathsf{pk}_{i_0}, \mathsf{pk}_{i_1}\})$.

In these security experiments the registration and corruption oracles $\mathcal{JO}$ and $\mathcal{CO}$ are removed to better reflect the *ad hoc* ring construction. Instead, arbitrary key input to the $\mathcal{SO}^{\mathsf{ACK}}$ and $\mathcal{LOR}^{\mathsf{ACK}}$ oracles is allowed and provide alternatives to the corruption oracle. The same modification can be made for the other properties in a similar manner. We depict the alternative experiments in Figure 3.2 on the facing page.

Definition 25: Linkable Ring Signature in the Adversarially-chosen Key Model

A *Linkable Ring Signature* scheme is defined with algorithms described in Definition 23 on page 54 and achieves a security in the *adversarially-chosen key model* if it achieves the properties of *Unforgeability* unf-ACK, *Anonymity* ano-ACK, *Linkability* link-ACK and *Non-slanderability* slan-ACK, described in Section 3.2 but, this time, on the basis of the experiments provided in Figure 3.2 on the facing page.

Most linkable ring signatures have been proposed without regard to this model (see the other works in Table 3.1 on page 49), although we believe that most linkable ring signatures could achieve this stronger properties in the adversarially-chosen key model, as it is not much more demanding on the design than the honest key model. Only two schemes in [BDH+19] and [BKP20] stand out from the rest of the literature and have been shown to be secure within the framework of this model. Further work is needed to re-examine the security of existing schemes in these models. Table 3.1 on page 49, column named *Anonymity* and Section 3.7 provide a literature review of the anonymity of the existing linkable ring signatures. Their we try to provide arguments towards the potential achievement of anonmity ano by most of the schemes of the literature.

3.5 Insecurity of the One-time Anonymity

Technical Summary

We set out our main concerns about the modelling of signer's anonymity 1-ano in Section 3.2. Despite this, it is the model used by almost all existing works. In this section, we present two counter-examples showing that this definition lacks anonymity. Our first counter-example is a dedicated scheme, while the second comes from an existing work [BL16] which has different purposes. Both show the need to adopt a stricter definition of anonymity, as after the second signature the identity of the signer is purposely revealed. Nevertheless, these constructions are

secure linkable ring signatures in the model of Section 3.2. This model was used to demonstrate the security of 16 linkable ring signatures out of the 18 existing schemes.

3.5.1 Toy Counter-example Scheme.

We start our dedicated construction from a secure linkable ring signature LRS, such any of the ones exposed in Table 3.1 on page 49. From this LRS we instantiate a new linkable ring signature scheme CeLRS for *Counter-example linkable ring signature*, by combining LRS with a secret sharing scheme $(\mathsf{Split}, \mathsf{Recover})$ (Definition 17 on page 38).

$\mathsf{CeLRS.Setup}_{\mathsf{LRS}}(1^\lambda)$**:** corresponds to the execution of $\mathsf{LRS.Setup}_{\mathsf{LRS}}(1^\lambda)$.

$\mathsf{CeLRS.Gen}_{\mathsf{LRS}}(1^\lambda)$**:** executes $(\mathsf{sk}_{\mathsf{LRS}}, \mathsf{pk}_{\mathsf{LRS}}) \leftarrow \mathsf{LRS.Gen}_{\mathsf{LRS}}(1^\lambda)$ and $s_1, s_2 \leftarrow \mathsf{Split}(\mathsf{pk}_{\mathsf{LRS}}, 2)$. Sets and returns $\mathsf{sk} = (\mathsf{sk}_{\mathsf{LRS}}, s_1, s_2)$, $\mathsf{pk} = \mathsf{pk}_{\mathsf{LRS}}$.

$\mathsf{CeLRS.Sign}_{\mathsf{LRS}}(\mathsf{sk}_i, m, \{\mathsf{pk}_j\}_{j\in\mathcal{R}})$**:** parses sk_i into $(\mathsf{sk}_{\mathsf{LRS}}, s_1, s_2)$, randomly samples $b \xleftarrow{\$} \{1, 2\}$ and returns $\sigma_{\mathsf{LRS}} \leftarrow \mathsf{LRS.Sign}_{\mathsf{LRS}}(\mathsf{sk}_{\mathsf{LRS}}, m\|s_b, \{\mathsf{pk}_j\}_{j\in\mathcal{R}})$ and s_b as σ.

$\mathsf{CeLRS.Verif}_{\mathsf{LRS}}(m, \sigma, \{\mathsf{pk}_j\}_{j\in\mathcal{R}})$**:** parses σ into σ_{LRS} and s. Executes $\mathsf{LRS.Verif}_{\mathsf{LRS}}(m\|s, \sigma_{\mathsf{LRS}}, \{\mathsf{pk}_j\}_{j\in\mathcal{R}})$ and returns its result.

$\mathsf{CeLRS.Link}_{\mathsf{LRS}}(\sigma, \sigma')$**:** parses σ into σ_{LRS} and s, and σ' into σ'_{LRS} and s'. Executes and returns the result of $\mathsf{LRS.Link}_{\mathsf{LRS}}(\sigma_{\mathsf{LRS}}, \sigma'_{\mathsf{LRS}})$.

The secret sharing share included in a single signature does not reveal any information about the signer's public key, since the secret sharing scheme is perfectly secret. On the other hand, we have considered a LRS scheme with one-time anonymity, which therefore does not reveal the identity of the signer. Nevertheless, a signer has a probability of at least $1/2$ of revealing its identity when it sends its second signature. Since one-time anonymity is modelled by a single signature disclosed to the adversary (see Section 3.2), this construction is proved secure as per Property 3 below and its proof. Moreover, the disclosure of the identity of the signer when more signatures can be claimed does not affect the other properties. This highlights limitations of the one-time anonymity 1-ano property in ensuring the hiding of the identity of the signer to its first signature for all linkable ring signatures of Table 3.1a.

Property 3

Consider a secure linkable ring signature LRS and a secret sharing scheme with perfect secrecy. Then, the above toy counter-example scheme CeLRS is a linkable ring signature with correctness, unforgeability unf, one-time anonymity 1-ano, linkability link and non-slanderability slan under the definitions introduced in Section 3.2 in any of the corruption models HK or ACK.

Proof. The correctness is straightforward. To give an intuition of the following argument, anonymity of the CeLRS construction follows from the anonymity of the LRS and the perfect secrecy of the secret sharing scheme. The other properties of the CeLRS

construction uniquely follow based on the security of the LRS scheme which has already been proven.

*Unforgeability (*unf*).* First, it should be noted that the LRS scheme is assumed to satisfy the unforgeability unf prescribed in Section 3.2, and that the share s_b is signed with the message. An adversary modifying s_b in the signature would cause the verification to fail because the wrong message would be introduced into the verification algorithm. Hence, the property follows from a direct reduction to unf of the LRS signature. A forgery against the CeLRS scheme for a message m would correspond to a forgery for a message $m\|s$ for a random s amongst s_1 or s_2.

*One-time Anonymity (*1-ano*) (unconditional if unconditional for the* LRS*).* This is a two step proof. As only one signature is provided to the adversary for the public identities pk_{i_0} and pk_{i_1}, the first step is to replace the element s embedded in the signature σ by a random element based on the perfect secrecy, this is possible as one of the shares is never disclosed during the experiment. From then on, the signature σ of the CeLRS construction is just a LRS signature with a random elements concatenated to the signed message. The one-time anonymity 1-ano of the LRS scheme guarantees that no identity related information would leak from the signature σ_{LRS}, hence from the signature σ provided to the adversary.

*Linkability (*link*) and Non-slanderability (*slan*).* As the linking algorithm only take into account the sub-signatures $\sigma_{\mathsf{LRS}_0}^*$, $\sigma_{\mathsf{LRS}_1}^*$, these experiments give the same answers for the CeLRS construction and the LRS scheme used as its base. Hence, linkability link and non-slanderability slan are both ensured under the hypothesis that the LRS scheme is secure. □

3.5.2 Model of k-Times Full Traceable Ring Signatures

This section recalls the model for *k-Times Full Traceable Ring Signature* originally introduced by Bultel and Lafourcade [BL16]. Their construction is a linkable ring signature that can be traced back to the signer when it produces more than k authorised signatures. We define it here as it is used in Section 3.5 to show that the 1-time full traceable ring signature presented in [BL16] can be demonstrated secure under the model of linkable ring signature with one-time anonymity 1-ano ilustrated in Section 3.2 despite the fact that it explicitly discloses the identity of the signer on the second signature. We chose to present this construction, we could also have presented the same arguments for the traceable ring signature in [FS07a].

Definition 26: k-Times Full Traceable Ring Signature (k-FTRS)

A *k-Times Full Traceable Ring Signature* scheme is composed of five algorithms defined as follows:

$\mathsf{Setup}_{\mathsf{k\text{-}FTRS}}(1^\lambda)$**:** is a PPT algorithm that takes the security parameter λ and produces the *public parameters* pp.

We assume these parameters pp as common input to all the following algorithms.

$\mathsf{Gen}_{\mathsf{k\text{-}FTRS}}(1^\lambda, k)$**:** is a PPT algorithm that takes the security parameter λ and a *threshold value* k denoting the maximum number of anonymous signatures authorised, it returns a pair of keys $(\mathsf{pk}, \mathsf{sk})$.

$\mathsf{Sign}_{\mathsf{k\text{-}FTRS}}(\mathsf{sk}_i, m, \{\mathsf{pk}_j\}_{j\in\mathcal{R}}, l)$**:** is a PPT algorithm that takes a *vector* $\{\mathsf{pk}_i\}_{i\in\mathcal{R}}$ of public keys for a ring set $\mathcal{R}$, a signer secret key sk_i (with $i \in \mathcal{R}$), a the witness $l \in \{1, \ldots, k\}$ and a message m. It outputs a ring signature σ.

$\mathsf{Verif}_{\mathsf{k\text{-}FTRS}}(m, \sigma, \{\mathsf{pk}_i\}_{i\in\mathcal{R}})$**:** is a deterministic polynomial-time algorithm that takes a public key vector $\{\mathsf{pk}_i\}_{i\in\mathcal{R}}$, a signature σ, and a message m, if the signature σ is valid, it returns 1, else it returns 0.

$\mathsf{Link}_{\mathsf{k\text{-}FTRS}}(\sigma, \sigma')$**:** is a deterministic polynomial-time algorithm that takes two signatures σ and σ', it returns 1, if they are linked, otherwise, it returns 0. Before running this algorithm, both signatures must be verified.

$\mathsf{Match}_{\mathsf{k\text{-}FTRS}}(\sigma, \sigma')$**:** is a deterministic polynomial-time algorithm that takes two signatures σ and σ', if $\mathsf{Link}_{\mathsf{k\text{-}FTRS}}(\sigma, \sigma') = 1$, it returns the public key of the signer pk and a *tracing element* ω, else it returns $\perp$.

$\mathsf{Trace}_{\mathsf{k\text{-}FTRS}}(\sigma, \omega)$**:** is a deterministic polynomial-time algorithm that takes a signature σ and a *tracing element* ω, it returns 1 if the signature σ has been produced by the signer associated to the tracer ω, else it returns 0.

A k-times full traceable ring signature k-FTRS must satisfy the properties of *Correctness*, *k-Unforgeability*, *k-Anonymity* and *k-Traceability*.

k-Unforgeability: constructing a valid signature without using the secret key should be unfeasible. The probability $\mathsf{Adv}^{\mathsf{k\text{-}unf}}_{\mathcal{A},\mathsf{k\text{-}FTRS}}(1^\lambda, k, n)$ of a PPT adversary $\mathcal{A}$ winning against the experiment $\mathsf{Exp}^{\mathsf{k\text{-}unf}}_{\mathcal{A},\mathsf{k\text{-}FTRS}}(1^\lambda, k, n)$ should be negligible for any integer $n \in \mathbb{N}$, any $k \leq n$ and any security parameter λ.

$\mathsf{Exp}^{\mathsf{k\text{-}unf}}_{\mathcal{A},\mathsf{k\text{-}FTRS}}(1^\lambda, k, n)$ - (Unforgeability experiment for k-FTRS)

$\mathsf{pp} \leftarrow \mathsf{Setup}_{\mathsf{k\text{-}FTRS}}(1^\lambda)$

$\{\mathsf{pk}_i, \mathsf{sk}_i\}_{1\leq i\leq n} \leftarrow \mathsf{Gen}_{\mathsf{k\text{-}FTRS}}(1^\lambda, k)$

$(m^*, \sigma^*, (\mathsf{pk}_i)_{i\in\mathcal{R}^*}) \leftarrow \mathcal{A}^{\mathsf{k}\mathcal{SO}_1}(\mathsf{pp}, (\mathsf{pk}_i)_{1\leq i\leq n})$

if $\mathcal{R} \not\subset \{1, \ldots, n\}$: **return** 0 // No corrupted public keys in the ring.

if $\sigma^* \notin \mathsf{k}\mathcal{SO}_1$: **return** 0 // The signature $\sigma^*$was not output by $\mathsf{k}\mathcal{SO}_1$.

return $\mathsf{Verif}_{\mathsf{k\text{-}FTRS}}(m^*, \sigma^*, \{\mathsf{pk}_i\}_{i\in\mathcal{R}^*}) = 1$

In this experiment, $\mathsf{k}\mathcal{SO}_1$ is a signing oracle that takes $(\mathsf{pk}_i, \{\mathsf{pk}_j\}_{j\in\mathcal{R}^*}, m, l)$ as input to sign the message m. If $\mathsf{pk}_i \notin \{\mathsf{pk}_i, \mathsf{sk}_i\}_{1\leq i\leq n}$ then it returns $\perp$, else it computes $\sigma \leftarrow \mathsf{Sign}_{\mathsf{k\text{-}FTRS}}(\mathsf{sk}_i, m, \{\mathsf{pk}_j\}_{j\in\mathcal{R}}, l)$ and returns σ.

k-Anonymity: guessing the public key corresponding to the secret key used to produce less than $(k+1)$ signatures should be hard. Any PPT adversary $\mathcal{A}$ should have a negligible advantage to win the the experiment $\mathsf{Exp}^{\mathsf{k\text{-}ano}}_{\mathcal{A},\mathsf{k\text{-}FTRS}}(1^\lambda, k, n)$:

$$\mathsf{Adv}^{\mathsf{k\text{-}ano}}_{\mathcal{A},\mathsf{k\text{-}FTRS}}(1^\lambda, k, n) = |\Pr[\mathsf{Exp}^{\mathsf{k\text{-}ano}}_{\mathcal{A},\mathsf{k\text{-}FTRS}}(1^\lambda, k, n) = 1] - 1/2| \leq \epsilon(1^\lambda),$$

for any integer $n \in \mathbb{N}$, any $k \leq n$ and any security parameter λ.

$\mathsf{Exp}^{\mathsf{k\text{-}ano}}_{\mathcal{A},\mathsf{k\text{-}FTRS}}(1^\lambda, k, n)$ - (Anonymity experiment for k-FTRS)

$b \xleftarrow{\$} \{0,1\}$
$\mathsf{pp} \leftarrow \mathsf{Setup}_{\mathsf{k\text{-}FTRS}}(1^\lambda)$
$\{\mathsf{pk}_i, \mathsf{sk}_i\}_{i=1}^n \leftarrow \mathsf{Gen}_{\mathsf{k\text{-}FTRS}}(1^\lambda, k)$
$(m^*, i_0, i_1) \leftarrow \mathcal{A}^{\mathsf{k}\mathcal{SO}_2}(\mathsf{pp}, \{\mathsf{pk}_i\}_{i=1}^n)$
$\sigma_0 \leftarrow \mathsf{k}\mathcal{SO}_2(m, \{\mathsf{pk}_j\}_{j\in\mathcal{R}}, \mathsf{sk}_{i_0}, l)$
$\sigma_1 \leftarrow \mathsf{k}\mathcal{SO}_2(m, \{\mathsf{pk}_j\}_{j\in\mathcal{R}}, \mathsf{sk}_{i_1}, l)$
$b^* \leftarrow \mathcal{A}^{\mathsf{k}\mathcal{SO}_2}(\sigma_b)$
return $b = b^*$

In this experiment $\mathsf{k}\mathcal{SO}_2$ is a signing oracle that takes $(\mathsf{pk}_i, \{\mathsf{pk}_j\}_{j\in\mathcal{R}}, m, l)$ in input to sign the message m. If $l > k$ or $\mathsf{pk}_i \notin \{\mathsf{pk}_j, \mathsf{sk}_j\}_{j=1}^n$ then it returns $\perp$ and aborts. If $l \in \{1, \ldots, k\}$ was already queried for pk_i, it also returns $\perp$. Else, it computes $\sigma \leftarrow \mathsf{Sign}_{\mathsf{k\text{-}FTRS}}(\mathsf{sk}_i, m, \{\mathsf{pk}_j\}_{j\in\mathcal{R}}, l)$ and returns σ.

k-Traceability: more then k signatures coming from the same signer are always (linkable and then) traceable. The probability $\mathsf{Adv}^{\mathsf{k\text{-}trace}}_{\mathcal{A},\mathsf{k\text{-}FTRS}}(1^\lambda, k, n)$ of a PPT adversary $\mathcal{A}$ winning against the experiment $\mathsf{Exp}^{\mathsf{k\text{-}trace}}_{\mathcal{A},\mathsf{k\text{-}FTRS}}(1^\lambda, k, n)$ should be negligible for any integer $n \in \mathbb{N}$, any $k \leq n$ and any security parameter λ.

$\mathsf{Exp}^{\mathsf{k\text{-}trace}}_{\mathcal{A},\mathsf{k\text{-}FTRS}}(1^\lambda, k, n)$ - (Traceability experiment for k-FTRS)

$\mathsf{pp} \leftarrow \mathsf{Setup}_{\mathsf{k\text{-}FTRS}}(1^\lambda)$
$\{\mathsf{pk}_i, \mathsf{sk}_i\}_{1\leq i\leq n} \leftarrow \mathsf{Gen}_{\mathsf{k\text{-}FTRS}}(1^\lambda, k)$
$i^* \leftarrow \mathcal{A}^{\mathsf{k}\mathcal{SO}_1}(\mathsf{pp}, \{\mathsf{pk}_i\}_{1\leq i\leq n})$
$(\{\mathsf{pk}_j\}_{j\in\mathcal{R}_i^*}, m_i^*, \sigma_i^*)_{1\leq i\leq l} \leftarrow \mathcal{A}^{\mathsf{k}\mathcal{SO}_1}(\mathsf{sk}_{i^*})$
if $l \geq k \wedge (\forall i \in \{1, \ldots, k\}, \mathsf{Verif}_{\mathsf{k\text{-}FTRS}}(m_i^*, \sigma_i^*, \{\mathsf{pk}_j\}_{j\in\mathcal{R}_i^*}) = 1$
$\wedge (\{\mathsf{pk}_j\}_{j\in\mathcal{R}_i^*}, m_i^*, \sigma_i^*) \notin \mathsf{k}\mathcal{SO}_1) \wedge \big((\forall 1 \leq a < b \leq k, \mathsf{Link}_{\mathsf{k\text{-}FTRS}}(\sigma_a, \sigma_b) \neq 1)$
$\vee (\exists a, b, i, \mathsf{Match}_{\mathsf{k\text{-}FTRS}}(\sigma_a, \sigma_b) = (\mathsf{pk}, \omega), \mathsf{pk} \neq \mathsf{pk}_{i^*} \vee \mathsf{Trace}_{\mathsf{k\text{-}FTRS}}(\sigma_i, \omega_i) \neq 1)\big)$
return 1
return 0

3.5.3 Concrete Counter-example

We present a second counter-example based on a construction which has been designed for a different purpose: revealing the public identity of signers overpassing a limit of k signatures. Originally proposed in [BL16], this primitive is called *k-times full traceable ring signature*. It is a ring signature that becomes linkable when the signer exceeds its limit of k allowed signatures. Once this limit has been exceeded, any verifier is capable of tracing the identity of the signer using an algorithm $\mathsf{pk} \leftarrow \mathsf{Trace}(\sigma, \sigma')$.

Here, we only consider *1-time full traceable ring signatures*, which allow a signer to produce one ring signature before disclosing their public key. Under the definition currently in use, we claim that this type of signature is also a linkable ring signature,

even though a linkable ring signature should not reveal the identity of the signer, even after an arbitrary number of issued signatures. We now examine the instance of the scheme from [BL16] with $k = 1$.

$\mathsf{Setup}_{\mathsf{LRS}}(1^\lambda)$**:** generates three groups $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t$ of prime order p with a pairing mapping $e\colon \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_t$ (a computable non-degenerate bilinear map). Picks up six generators $g_1, h_0, h_1, h_2, h_3 \in \mathbb{G}_1$ and $g_2 \in \mathbb{G}_2$ and a hash function H mapping to $\mathbb{Z}_p^*$.

$\mathsf{Gen}_{\mathsf{LRS}}(1^\lambda)$**:** the keys come in two parts, two secret discrete logarithms x and y constitute the secret key sk and the two associated elements of $\mathbb{G}_1$, $\mathsf{pk}_1 = g_1^x$ and $\mathsf{pk}_2 = g_1^y$ constitute the public key pk.

$\mathsf{Sign}_{\mathsf{LRS}}(\mathsf{sk}_i, m, \{\mathsf{pk}_i\}_{i\in\mathcal{R}})$**:** samples a random $r \xleftarrow{\$} \mathbb{Z}_p^*$ and computes $u = H(m, 0, g_2^r)$, $v = H(m, 1, g_2^r)$, $T_1 = h_1^y$, $T_2 = h_2^y \cdot g_1^{u\cdot x}$, $T_3 = h_3^y \cdot h_4^{v\cdot x}$, $T_4 = g_2^r$, $T_5 = e(h_4, T_4)^x$. Generates a zero-knowledge proof to wrap up all the elements:

$$\Pi \leftarrow \mathsf{ZK}\left\{ x, y, r : \begin{array}{l} \left(\bigvee_{(\mathsf{pk}_1,\mathsf{pk}_2)\in\{\mathsf{pk}_i\}_{i\in\mathcal{R}}} (\mathsf{pk}_1 = g_1^x \wedge \mathsf{pk}_2 = g_1^y)\right) \\ \wedge T_1 = h_1^y \wedge T_2 = h_2^y \cdot g_1^{u\cdot x} \wedge T_3 = h_3^y \cdot h_4^{v\cdot x} \\ \wedge T_4 = g_2^r \wedge T_5 = e(h_4, T_4)^x \end{array} \right\}.$$

Returns $\sigma = (T_1, T_2, T_3, T_4, T_5, \Pi)$ as the signature of the message m.

$\mathsf{Verif}_{\mathsf{LRS}}(m, \sigma, \{\mathsf{pk}_i\}_{i\in\mathcal{R}})$**:** parses the signature, computes $u = H(m, 0, T_4)$, $v = H(m, 1, T_4)$ and verifies the zero-knowledge proof.

$\mathsf{Link}_{\mathsf{LRS}}(\sigma, \sigma')$**:** parses the signatures, checks if $T_1 = T_1'$ and returns 1 if so, otherwise returns 0. We assume that the signatures have been verified before.

For the sake of completeness and the rest of our argument, we also provide the tracing algorithm that identify signers who have produced more than one signature. It also encompass the matching algorihm Match directly inside the tracing algorithm Trace.

$\mathsf{Trace}(\sigma, \sigma')$**:** checks the link between the two signatures by executing $\mathsf{Link}_{\mathsf{LRS}}(\sigma, \sigma')$ and stop if it fails. On two signatures σ, σ' being linked, computes u, u' and $\mathsf{id} = (T_2/T_2')^{1/(u-u')}$. Returns the identity id. A second element $w = (T_3/T_3')^{1/(v-v')}$ is also recovered in the construction presented in [BL16], this element is not useful in the case $k = 1$.

Property 4

The 1-time full traceable ring signature from Bultel and Lafourcade introduced in [BL16] and depicted above is a linkable ring signature and achieves correctness, unforgeability unf, one-time anonymity 1-ano, linkability link and non-slanderability slan under the definitions introduced in Section 3.2 in any of the corruption models HK or ACK.

Proof. Correctness is straightforward. Security proofs for the scheme are given for the associated model in [BL16], we rely on them to construct ours. Indeed, their model is quite similar to the security model of linkable ring signatures.

Unforgeability. The experiment unf presented in Section 3.2 matches the k-unf experiment in [BL16] (recalled in Section 3.5.2): the adversary has to return a valid signature, *i.e.,* $\mathsf{Verif}_{\mathsf{LRS}}(m^*, \sigma^*, \{\mathsf{pk}_i\}_{i\in\mathcal{R}}) = 1$, for a set of uncorrupted users with honestly generated keys $\{\mathsf{pk}_i\}_{i\in\mathcal{R}}$. Furthermore, this signature should not have been output by a call to the signature oracle. Thus, unforgeability unf is obtained directly from the proof of unforgeability k-unf given in [BL16]. It relies mainly on the soundness of the zero-knowledge proof Π.

One-time Anonymity. The experiment $\mathsf{Exp}^{\mathsf{k\text{-}ano}}_{\mathcal{A},\mathsf{k\text{-}FTRS}}(1^\lambda)$ (recalled in Section 3.5.2) is stronger than the one-time anonymity 1-ano introduced for the LRS schemes and presented in Section 3.2. In their model, the authors of [BL16] made it possible for the adversary to obtain multiple signatures from the same designated signer, with a limit of k signatures per signer. Here we have set the limit $k = 1$ which directly provides 1-ano. The model from [BL16] works in a static framework: the number of public keys in the ring is fixed to an integer n. Reducing our case to the static environment implies the introduction of a polynomial factor in the reduction. Consequently, k-ano implies one-time anonymity 1-ano.

Linkability. We are looking at the construction proposed in [BL16] using the setup $k = 1$, where k denotes the number of signatures that can be produced without being traced (linkability of all the produced signatures and identification of authors of the signatures). As the adversary can infer the identity of signers through its calls to the signing oracles, the ability to trace does not reveal any information. Hence, this property falls under the *traceability* of the 1-time full traceable ring signature as only one signature is queried for the challenger signers. Moreover, the proof Π is sound under the hardness of the DL problem (see [BL16] for the security proof). Since the adversary is unable to forge a signature for an honest and uncorrupted user under the soudness of π, linkability is an implication of the correctness of the Link algorithms.

Non-slanderability. In this experiment the condition $\mathsf{Link}_{\mathsf{LRS}}(\sigma, \sigma^*) = 1$ enforces that $T_1 = h_1^y = h_1^{y^*} = T_1^*$, hence $y = y^*$, where y and y^* are such that for $\mathsf{pk} = (\mathsf{pk}_1, \mathsf{pk}_2)$ and $\mathsf{pk}^* = (\mathsf{pk}_1^*, \mathsf{pk}_2^*)$, $\mathsf{pk}_2 = g_1^y$ and $\mathsf{pk}_2^* = g_1^{y^*}$. Under the soundness of the proof Π, the adversary must know y^* (thus y too). And under the zero-knowledge property of Π, the security of the against non-slanderability is reduced to the hardness of the DL problem. □

Therefore, the 1-time full traceable ring signature of [BL16] can also be considered as a linkable ring signature secure under the 1-ano -HK model presented in Section 3.2. From these counter-examples, we have demonstrated the existence of a gap between the definition of anonymity provided in most of the literature and the informal and expected purposes of this property. We now provide the formalism which has only been used by [BDH+19, BKP20]. Later, in Section 3.6, we demonstrate that these definitions bridge the anonymity gap in the definition.

3.6 Review of our Counter-examples

In this section, we evaluate the anonymity of our counter-examples.

Since it is now possible to obtain multiple signatures of the challenger signer based on the $\mathcal{LOR}$ oracle, our two counter-examples have become insecure for the new definition of anonymity. This is because a PPT adversary can claim more than one signature for one of the challenger signers when interacting with the challenger in one of the two $\mathsf{Exp}^{\mathsf{ano}}_{\mathsf{LRS}}(1^\lambda)$ experiments. As shown in Section 3.5, both schemes have non-negligible probabilities of revealing the identity of their signer after the second signature. For our counter-example construction, the probability of obtaining both secret sharing shares after the second signature is $1/2$, which allows identity recovery with a probability significantly different from $1/2$ (random guessing) even if we had obtained unconditional one-time anonymity based on an unconditionally anonymous LRS and a perfectly secret secret sharing scheme. This shows that even some schemes with unconditional one-time anonymity 1-ano may reveal the identity of a signer after its second signature.

Regarding Bultel and Lafourcade's 1-time full traceable ring signature [BL16], their primitive is specifically designed to reveal the identity of the signer after a given number of signatures. We have set this value to 2 in Section 3.5. Thus, given a polynomial number of queries to the signature oracle, an adversary can always query the oracles twice and break the game by revealing the identity of the signer based on the Trace algorithm. The arguments above show that our two counter-examples cannot achieve the definitions of anonymity of Section 3.4 and this holds even under the honest key model provided in Section 3.3. The above arguments imply the following property.

Property 5

Our *counter-example* of Section 3.5.1 and the 1-time full traceable ring signature from [BL16] do not guarantee anonymity ano in the adversary-chosen keys model, nor in the honest keys model.

3.7 Literature Review

Technical Summary

The results of these investigations regarding expected security, obtained after a broad review of the literature, are summarized in Table 3.2 on the facing page. We are expecting that most schemes verify the stronger definitions of Section 3.4 as they were constructed with this idea in mind, while schemes in [BDH+19] and [BKP20] have already been proven secure by their authors in the model of Section 3.4. Furthermore, most security reductions of existing schemes were provided based on arguments applied to one signature and decorrelating it from the keys of the signer. Their security proofs, for most of them, can be generalised when these arguments can be applied independently using hybrid arguments. This is unlike the reduction we provided for our counter-examples[3].

Reference	One-time Anonymity (1-ano)	Anonymity (ano)
Liu *et al.* [LWW04]	Computational	✗ → ✓
Tsang *et al.* [TWC+05]	Computational	✗ → ✓
Liu and Wong [LW05]	Computational	✗ → ✓
Tsang and Wei [TW05]	Computational	✗ → ✓
Liu *et al.* [LASZ13]	Unconditional	✗ → ✓ (Unconditional)
Yuen *et al.* [YLA+13]	Computational	✗ → ✓
Boyen and Haines [BH18]	Unconditional	?[4]
Branco and Mateus [BM18]	Computational	?[5]
Baum *et al.* [BLO18]	Computational	✗ → ✓
Lu *et al.* [LAZ19]	Computational	?[5]
Liu *et al.* [LNY+19]	Computational	No Proof Found
Zhang *et al.* [ZLS+20]	Computational	✗ → ✓
Balla *et al.* [BBG+22]	Unconditional	✗ → ✓(Unconditional)
Bootle *et al.* [BEHM22]	Computational	✗ → ✓
Xiangyu *et al.* [HC24]	Computational	✗ → ✓
Xue *et al.* [XLAZ24]	Computational	✗ → ✓

(a) Existing Linkable Ring Signatures. "✗ → ✓" means that it seems possible to extend the existing proof. N.A. for "Not Applicable".

Reference	One-time Anonymity (1-ano)	Anonymity (ano)
Alberto *et al.* [ATSS+18]	Unconditional	✗[5]

(b) Existing One-time Linkable Ring Signatures.

Reference	One-time Anonymity (1-ano)	Anonymity (ano)
Backes *et al.* [BDH+19]	N.A.	✓ (already proven)
Beullens *et al.* [BKP20]	N.A.	✓ (already proven)

(c) Existing Linkable Ring Signatures with Proven Anonymity. N.A. for "Not Applicable".

Table 3.2: Anonymity of Existing Linkable Ring Signatures.

In this section, we provide a systematic review of all existing schemes in the literature in term of the experiment introduced in Section 3.4. Given the arguments of Section 3.5 and 3.6, it becomes apparent that the security of linkable ring signatures with one-time anonymity 1-ano should be re-evaluated, even when one-time anonymity holds unconditionally. We give an overview of how the stronger security requirement of anonymity applies to existing systems. Yet, we do not seek to prove the security of existing schemes.

Given their design choices, it seems that the authors of the schemes in the literature aimed to offer the security described in the stronger model, when one-time anonymity was not considered to be a feature of the scheme. Indeed, this is reflected in the informal description of anonymity provided in previous works. We stress however that, even if the quoted schemes seem to have been designed to achieve our security, it would be necessary to re-analyse their security in the model of Section 3.4.

All linkable ring signatures include the ability to link signatures generated from the secret key sk. In general, these signatures can be divided into two parts: σ the "signature" itself and a *tag*. The purpose of the *tag* is to link valid signatures by their

[3] Our counter-examples where purposely lacking security when more signature needed to be produced, but their reduction involved arguments that were not limited to a single signature each time (*e.g.,* the perfect secrecy of the Sharmir secret sharing apply to $k-1$ shares but does not holds anymore when the k^{th} shared is revealed).

[4] There is no direct argument one way or the other, we are leaving this question open.

[5] This scheme is a one-time LRS, in their case, one-time anonymity is expected.

direct comparison while being bound to the "signature" part. The tags are usually in the form $tag = h^{\mathsf{sk}}$, for some fixed element h when relying on DL related hypothesis, or similarly when relying on other mathematical bases. The "signature" part wraps everything together to avoid modification of the tag, it can for example be an "$\mathtt{OR}$" proof over the Schnorr NIZK proof [CDS94, FS87] over all the public keys $\mathsf{pk} = g^{\mathsf{sk}}$. This construction was studied in [TWC+05] with a proof in the model of Section 3.2. As part of the security reduction of the anonymity, these tags are being stripped of the signer's identity by applying decisional hypothesis, *e.g.,* the DDH hypothesis for tags formed as above, then, providing a random value g^z instead of $h^{\mathsf{sk}} = g^{x \cdot \mathsf{sk}}$ for some unknown x. The reduction for other parts of the signature is more specific to the design. We detail below existing lines of work and their methods.

General Idea of our Analysis. When investigating the anonymity proofs of existing signature schemes, it was common to be able to divide the proof into three parts: (1) an initial sequence of game hops, *e.g.,* programming the ROM, (2) a sequence involving the modification of elements limited to the signature part σ of the challenge decorrelating all but the tag from the signer's secret key, *e.g.,* simulation of the NIZK proof wrapping up the signature, (3) a sequence of game hops making it possible to decorrelate the signature tag of the signature σ from the signer's keys for example the one based on the DDH and mentioned above. Now, given steps 2 (associated with the challenge signature) and 3 (associated with the label value), a hybrid argument seems to be possible most of the time to apply independently these parts of the proof to the multiple challenges generated by the $\mathcal{LoR}$ oracle. In particular, the proof of [BKP20], for which the scheme is secure in the strong model of Section 3.4, mainly follows these steps. We therefore investigated whether it is possible to obtain a hybrid argument based on the reductions provided to decorrelate the multiple challenges of the signer's identity and summarised our results in Table 3.2 on the previous page.

Zero-knowledge Based LRS Schemes. As a prominent basis for LRS, the constructions from [TWC+05, LW05, TW05, YLA+13, BM18, BEHM22, HC24, XLAZ24] are based on zero-knowledge proofs, zero-knowledge arguments, or signature of knowledge. These schemes are used to wrap up ring signatures and link them with tags. The reductions provided by the authors of the existing schemes are mainly based on the zero-knowledge security of their NIZK proofs. This leads us to believe that the security of the previous schemes can be extended to anonymity ano in the adversarially-chosen keys model. This is because the proofs corresponding to the signature can be simulated independently and, by virtue of the existing proof for one-time anonymity, it must be possible to decorrelated the tag from the signer's keys. This last reduction for the tag most likely applies to several signatures at the same time.

Pedersen Commitment Based LRS Schemes for Unconditional Anonymity. The Pedersen commitment [Ped91] where two secret values r and s are sampled and form a public commitment $c = g^r h^s$, for two generators g and h of a group, was used to obtain unconditional anonymity for LRS. LRS scheme based on this commitment scheme uses the elements r and s as the secret key and c the public key. As multiple

pairs (r, s) leads to the same public key pk, an unbounded adversary is unable to recover the secret from the public key. The anonymity reductions provided by the authors of these schemes [LASZ13, BH18, BBG+22] are essentially the same. For any signature, there is always a secret key pair leading to any public key involved and from which the same signature could result. Put differently, whatever secret key is used, the statistical distribution of the signature remains unchanged. Given the independence of the signatures from the secret keys, we claim that the proof for all three schemes can be generalised to prove the stronger notion of anonymity ano, at least under the honest key model.

Remaining LRS Schemes. Among the existing schemes, some do not fall into the previous two categories and their anonymity relies solely on decisional hypotheses, such as the DDH problem for [LWW04, ZLS+20] or the *Decisional Module-LWE problem* [BDK+18] for [BLO18] and [LNY+19]. Another scheme [LAZ19] is based on the chameleon hash function. For the first two schemes [LWW04, ZLS+20], each of the provided arguments only apply to a single element in the signatures and the associated reductions can be performed an arbitrary polynomial number of times. We therefore believe that a hybrid argument could be carried out based on part of the provided reduction and generalise the proof for any number of signatures produced by a single signer. That is why anonymity ano seems possible to guarantee. As for the scheme [LNY+19], we cannot verify how the reduction is performed for this scheme as we were unable to find an obvious reference to the full proofs.

The security of the remaining [ATSS+18] scheme does not need to be addressed because its authors have proposed a singleone-time linkable ring signature: a LRS that is intended to be used to produce a single signature for each generated key pair. In particular, this scheme is unforgeable only if a single signature has been produced with a key pair. There is therefore no need to consider more than one signature for each key pair in the anonymity experiment.

3.8 Relationship Between the Properties

All the relationships between the four anonymity properties are shown in Figure 3.3. Some of them have not yet been demonstrated, and we discuss them below.

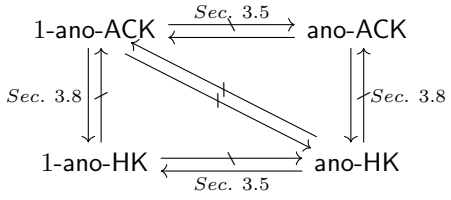


Figure 3.3: Comparison of the Anonymity Levels in the Various Corruption Models.

The anonymity ano presented in Section 3.4 (*resp.* in Section 3.3) in the ACK corruption model (*resp.* the HK) is stronger than the anonymity 1-ano, as it allows access to several challenge signatures whereas only one is provided to the adversary in the 1-ano-ACK model (*resp.* 1-ano-HK). This has been demonstrated in Section 3.5.

Now, we examine the relationship between the HK corruption model and the ACK corruption model. Consider an adversary $\mathcal{A}$ winning, with non-negligible probability, against the experiment 1-ano or ano in the honest key model HK. According to the prescriptions of the honest key model, in the case of experiment 1-ano, $\mathcal{A}$ did not query oracle $\mathcal{SO}$ and did not issue a public key vector $(\mathsf{pk}_i)_{i\in\mathcal{R}^*}$ with an unregistered or corrupted public key. Similarly for the case of the ano experiment, $\mathcal{A}$ did not query oracle $\mathcal{SO}$ or oracle $\mathcal{LOR}$ with unregistered or corrupted public keys. Hence, the same answer provided in the ACK model would also be accepted by the respective decisional problems in the ACK model. Therefore, 1-ano-HK is weaker than the 1-ano-ACK experiment and ano-HK is weaker than the ano-ACK experiment. Let us now show that there is a scheme that achieves security in the HK model but not the ACK model. To do this, we provide a second toy scheme showing that the inequalities between the corruption models are strict.

Second Toy Counter-example Scheme. Consider a secure signature LRS in any of the corruption models and a IND-CPA secure encryption scheme $\mathcal{E}$. The following counter-example encrypts the signer's identity under all the other public encryption keys of the ring members. This allows anyone with the secret key associated with one of the public keys of any of the ring members to recover the identity of the signer, but not anyone outside the ring. We formalise below the linkable ring signature scheme with such a property and show that it fulfils all the properties of LRS schemes in the HK corruption model but not in the ACK corruption model.

$\mathsf{Setup}_{\mathsf{LRS}}(1^\lambda)$**:** corresponds to the execution of $\mathsf{LRS.Setup}_{\mathsf{LRS}}(1^\lambda)$.

$\mathsf{Gen}_{\mathsf{LRS}}(1^\lambda)$**:** executes $(\mathsf{sk}_{\mathsf{LRS}}, \mathsf{pk}_{\mathsf{LRS}}) \leftarrow \mathsf{LRS.Gen}_{\mathsf{LRS}}(1^\lambda)$ and $(\mathsf{sk}_{\mathcal{E}}, \mathsf{pk}_{\mathcal{E}}) \leftarrow \mathsf{Gen}_{\mathsf{Enc}}(1^\lambda)$. Sets and returns $\mathsf{sk} = (\mathsf{sk}_{\mathsf{LRS}}, \mathsf{sk}_{\mathcal{E}})$, $\mathsf{pk} = (\mathsf{pk}_{\mathsf{LRS}}, \mathsf{pk}_{\mathcal{E}})$.

$\mathsf{Sign}_{\mathsf{LRS}}(\mathsf{sk}_i, m, \{\mathsf{pk}_j\}_{j\in\mathcal{R}})$**:** parses sk_i into $(\mathsf{sk}_{\mathsf{LRS}}, \mathsf{sk}_{\mathcal{E}})$ and for all i in the ring $\mathcal{R}$ parses $\mathsf{pk}_j \xrightarrow{p} (\mathsf{pk}_{\mathsf{LRS},j}, \mathsf{pk}_{\mathcal{E},j})$. It computes $e_j \leftarrow \mathsf{Enc}(\mathsf{pk}_{\mathcal{E},j}, \mathsf{pk}_i)$ and returns $\sigma_{\mathsf{LRS}} \leftarrow \mathsf{LRS.Sign}_{\mathsf{LRS}}(\mathsf{sk}_i, m\|(e_j)_{j\in\mathcal{R}}, \{\mathsf{pk}_j\}_{j\in\mathcal{R}})$ and $(e_j)_{j\in\mathcal{R}}$ as σ.

$\mathsf{Verif}_{\mathsf{LRS}}(m, \sigma, \{\mathsf{pk}_j\}_{j\in\mathcal{R}})$**:** parses σ into σ_{LRS} and $(e_j)_{j\in\mathcal{R}}$ and verify σ_{LRS} by executing $\mathsf{LRS.Verif}_{\mathsf{LRS}}(m\|(e_j)_{j\in\mathcal{R}}, \sigma_{\mathsf{LRS}}, \{\mathsf{pk}_j\}_{j\in\mathcal{R}})$ and returns its result.

$\mathsf{Link}_{\mathsf{LRS}}(\sigma, \sigma')$**:** parses σ into σ_{LRS} and $(e_j)_{j\in\mathcal{R}}$, and σ' into σ'_{LRS} and $(e'_j)_{j\in\mathcal{R}'}$. Executes and returns the result of $\mathsf{LRS.Link}_{\mathsf{LRS}}(\sigma_{\mathsf{LRS}}, \sigma'_{\mathsf{LRS}})$.

Property 6

Consider a secure linkable ring signature LRS with one-time anonymity 1-ano (*resp.* anonymity ano) and a IND-CPA secure encryption scheme $\mathcal{E}$. Then, the second toy counter-example scheme is a linkable ring signature with correctness, unforgeability unf, one-time anonymity 1-ano (*resp.* anonymity ano), linkability link and non-slanderability slan under the honest key model HK.

Proof. This proof follows an analogous path to the proof of the first toy counter-example, the proof of Property 3 but relyies on the IND-CPA security of the encryption scheme

for the $|\mathcal{R}|$ encrypted elements e_j for $j \in \mathcal{R}$ instead of the perfect secrecy of the secret sharing scheme (for the element s_1 or s_2 in the proof of the first toy counter-example). We now only elaborte for the proof of the property of anonymity 1-ano-HK or ano-HK:

First, for all challenge signatures with a ring $\mathcal{R}$, we reduce to the IND-CPA security of the encryption scheme for all the elements e_j, for all $j \in \mathcal{R}$. This is possible as in the honest key model, the secret key associated to the public key used by the challenger to encrypt the signers' identities remains unknown by the adversary. After this reduction, all elements e_j, for any $j \in \mathcal{R}$ and for all $\mathcal{R}$ supplied by the adversary, are uniformly random. Thus, in a similar way to the proof of Property 3, the 1-ano-HK property (*resp.* the ano-HK property) of the LRS leads to the proof of the 1-ano-HK property (*resp.* the ano-HK property) of our second toy example. □

Our arguments are valid for both the 1-ano-HK or link-HK experiment depending on the anonymity of the LRS scheme. And yet, it is clear that the ACK corruption model allows neither. Therefore, we conclude that the HK model is strictly weaker than the ACK model as presented in Figure 3.3.

The combination of the two counter-examples introduced in this Section and in Section 3.5, directly implies that there is no hierarchy between the 1-ano-ACK experiment and the ano-HK model. If we take any 1-ano-ACK secure linkable ring signature scheme and introduce it into our first toy counter-example, we still get a 1-ano-ACK secure linkable ring signature scheme. However, this time we ensure that it does not achieve ano anonymity, and therefore does not reach ano-HK. Similarly, if we consider a ano-HK secure linkable ring signature scheme and introduce it into our second toy counter-example, we still obtain a ano-HK secure linkable ring signature scheme, but we ensure that it does not achieve any type of anonymity in the ACK model. With these last elements, we conclude the comparison introduced in Figure 3.3.

3.9 Conclusion of the Chapter

We have demonstrated that most security analyses for existing linkable ring signatures lacked of any guarantee of anonymity, even for the most recently proposed ones. To support our claim, we provided two constructions that can be proven secure under the most commonly used security model, despite clearly breaking the informal anonymity expected from such schemes. Indeed, these counter-examples leaked the identity of the signer after only two signatures.

On the basis of this observation, we highlighted the model proposed by Backes *et al.* [BDH+19] and subsequently used by Beullens *et al.* [BKP20] which has been left out of subsequent works. We believe that their model in the adversarially-chosen keys setting, better reflects the use cases of linkable ring signatures unlike the currently used one. In particular, they leave out the two counter-example constructions as we demonstrated.

Based on their model, we reviewed the literature providing arguments in favor of existing schemes realising the new properties. Thus, we rule out a global lack of anonymity for existing schemes.

Finally, for completeness, we have fully classified the two anonymity properties in both the honest key model and the adversarially-chosen keys model.

Chapter 4

k-Times Full Traceable Proxy and Sanitizable Signatures

Chapter Summary

This chapter introduces a method to achieve fully traceable k-times anonymity in anonymous signature schemes, particularly for proxy and sanitizable signatures. For each of these properties, we formalise the concept, propose a security model, and present a schema under the DDH assumption. These schemes are efficient due to their logarithmic key/signature size relative to k. Here are our main contributions:

- Achieve k-times full traceability anonymity for delegation-based signatures.
- Formalise the definition of algorithms, provide a security model and present a provably-secure scheme under the DDH assumption.
- Ensure that keys/signatures are logarithmic in k, making schemes practical even for large values of k.

Contents

4.1 Introduction to the Chapter Content

Proxy signatures [MUO96], enable a signer to delegate the ability to sign messages on its behalf to a delegate. This cryptographic primitive has attracted a great deal of interest in recent decades. In some contexts, it is preferable to hide the delegate's identity from the signature verifier. Such a signature is called an *anonymous proxy signature* [FP08]. A trivial way of achieving this property is to give the delegate the signing key directly; however, this technique allows the delegate to impersonate the signer without any constraint, which is clearly not desirable. The signer therefore needs a way of tracing its delegates if one of them abuses their power. This leads to two inherent issues: the signer must be active to manage the actions of its proxies, and must have access to the signatures.

The concept of traceable k-times anonymity offers an alternative way to delegate tracing. Signature schemes following this paradigm allow users to create k signatures anonymously. If they exceed this limit, a verifier can then publicly link two signatures and trace the identity of the signer. This property has been defined for *ring signatures* [FS07b], *group signatures* [ASY06], and *anonymous authentication* [TFS04]. Moreover, a k-times signature is said to be *fully traceable* when the verifier can retrieve all the signatures generated by the signer which has exceeded the k limit *a posteriori*. To the best of our knowledge, this more powerful property has only been defined for ring signatures [BL16] (more details are given in Section 3.5.2).

However, k-times anonymity has never been applied directly to proxy signatures, even though they seem naturally suited to this property. Achieving k-times anonymity would enable a verifier, which has access to all signatures, to publicly trace dishonest proxies on its own, while preserving the anonymity of honest proxies, without the intervention of the signer. In this chapter, we close this gap by modeling and instantiating the first k-times fully traceable anonymous proxy signatures. We have illustrate this concept in Figure 1.6a on page 17 before examining it in detail in Section 4.3 on page 88.

On the other hand, *sanitizable signatures* [ACDMT05] are conceptually close to proxy signatures: in this primitive, the delegate (called the sanitizer) can no longer produce signatures by itself, but can modify certain parts of a signed message. When considering a setting where the sanitizer must remain anonymous, the same problems arise as with proxy signatures. Applying a similar approach, we propose the first k-times fully traceable anonymous sanitizable signatures. We have illustrated this concept in Figure 1.6b on page 17 before examining it in detail in Section 4.4.

Contributions and Technical Overview. We give a formal definition, a security model, and an efficient scheme (in term of size of the keys/signatures) for fully traceable k-times anonymous proxy signatures and fully traceable k-times anonymous sanitizable signatures. We give security proofs for these schemes. From a technical point of view, we rely on the method proposed in [BL16]: the delegate has k different public/secret keys; if it reuses the same key twice, then it is possible to link the two signatures to the user and extract an element that links all its other signatures. However, this method requires a number of keys linear in k; our main technical contribution is a method for generating k distinct and mutually unlinkable keys from $2\log_2(k)$ keys only. The idea

is to compose, at the i^{th} signature, the keys corresponding to the bits of i to obtain a new public/secret key pair. These keys must be certified by the delegator, but must be unlinkable. To achieve these two properties simultaneously, we use a signature on equivalence class [FHS19], which allows the delegate to randomise the $2\log_2(k)$ keys while maintaining the validity of their certificate. This method requires the creation of an *ad hoc* zero-knowledge proof ensuring the verifier that the delegate has correctly generated its key. For the special case where k is not a power of 2, we build another *ad hoc* zero-knowledge range proof to ensure that i is indeed less than k. Both of these proofs have logarithmic complexity in size, enabling us to obtain logarithmic complexity in size for both our keys and our signatures. This method is fairly generic, so we think it could be of independent interest in other primitives requiring the generation of several certified keys. Our sanitizable signature scheme uses the same technique as the one proposed in [BB21], combined with the method described above to make it k-times anonymous. The main technical challenge here is to adapt the signature to enable the signer to simulate the use of the $2\log_2(k)$ keys in the original signature, so that it is not possible to determine whether it has been sanitized or not.

For each signature primitive, we define the following properties in addition to unforgeability:

Anonymity: signatures are anonymous as long as the delegate does not exceed k signatures. In particular, they cannot be linked to each other.

Traceability: if the delegate exceeds the k signatures limit, it cannot prevent anyone from linking all its signatures and recovering its identity.

Non-framability: a delegate cannot produce a signature that can be traced back to another entity.

We also adapt the security properties of sanitizable signatures:

Immutability: it is not possible to modify parts of messages that are not intended to be modified.

Transparency: it is not possible to guess whether a signature has been sanitized or not. This property implies the property of *privacy*: it is not possible to determine any information about the original message.

Unlinkability: it is not possible to link a sanitized signature to the original signature, or to link sanitized signatures from the same original signature. A few schemes, such as [FKM+16], achieve this property. Note that unlinkability differs from anonymity, which ensures that it is not possible to link signatures from the same user. We provide more details about this in Section 4.4.

Invisibilty: it is not possible to identify which part of the message is modifiable. Note that designing schemes that are both unlinkable and invisible is challenging, and there are only two schemes in the literature that combine these properties [BLL+19, BB21].

Finally, we informally discuss alternatives in terms of functionality and security for our protocols. As the ad hoc secret keys are generated by the delegator, it can always trace the signatures. This feature is desirable in most cases, but we note that it is possible to adapt our protocol to achieve full anonymity, even from the delegator's point of view. To achieve this, the proxy simply generates the $2\log_2(k)$ secret keys itself, and sends the corresponding public keys to the proxy to produce the certificate. In

return, the delegation process becomes interactive, whereas it is not when the delegator generates the keys.

On the other hand, we have chosen to define sanitizable signatures where the sanitizer is anonymous, and where the overall number of sanitizations (potentially applied to different signatures) is limited by k. We could also have defined a non-anonymous sanitizer and parameterized the primitive so that unlinkability is guaranteed as long as the sanitizer does not exceed the sanitization limit k for each signature. Note that the limit could be different for each signature. In this case, the primitive would have been k-times unlinkable and not k-times anonymous. This variant follows naturally from our construction: we only need to store an ad hoc delegation keys in each signature, using randomizable encryption.

Motivations. Anonymous proxy signatures are used wherever an entity wishes to delegate the ability to sign on its behalf to others, without making the delegation policy transparent to the recipient of the messages. Anonymity can also protect proxies when their identity must remain secret, for example in legal proceedings where retaliation is possible. Conversely, anonymity provides a high level of protection for proxies who might be tempted to abuse their power. The k-times fully traceable anonymity property can significantly limit this, even in the absence of the delegate. Sanitizable signatures extend proxy signatures by adding a degree of control over the messages sent by delegates. For example, they can be used to force the use of message templates.

For instance, consider a manager who delegates the ability to sign and send emails on their behalf from their email address to multiple entities. These could be employees or servers that automatically send emails that contain, depending on their role, specific messages, appointments, contracts, payments, invoices, reminders, summons or other legal or commercial documents. If too many emails are being sent from the same entity on behalf of the manager, the company's mail server can use the k-times mechanism to locate the offending entity, block the emails it is sending, list all its signatures and alert the manager and anyone else who has received emails from this entity in the past. Note that in our case the server is honest but curious: we trust it to check signatures and detect anomalies (it cannot be fully corrupted by an active attacker), but the information it processes does not allow it to learn anything about the identity of honest proxies or the delegation policy (a passive attacker can observe everything that passes through the server without compromising anonymity).

To control the content of messages, it is helpful to use sanitizable signatures that force delegates to use templates. For example, by setting the metadata it is possible to allow emails to be sent only to certain people, on certain dates, with certain subjects, or by forcing the addition of copy users who can check the content of the email. In the case of automatic emails, such as invoices, it is possible to impose a very precise template where only the customer's name, date and amount can be changed. Note that thanks to the security properties of our sanitizable signatures (transparency, anonymity, unlinkability, and invisibility), the company's delegation policy remains entirely private from the point of view of the verifiers and the mail server as long as the k limit is not exceeded.

Related work. Anonymous proxy signatures were introduced by Fuchsbauer and Pointcheval [FP08]. Since then, several other anonymous proxy signature schemes were proposed [WYM14, WYML15]. Unfortunatly, as mentioned above, they all consider active traceability management by the original signer or a dedicated semi-trusted proxy. Unlike our scheme, Fuchsbauer and Pointcheval's scheme allows hierarchical management of proxies (a delegate can allow a sub-delegate to sign in its place, etc.). This feature could naturally be achieved by extending our scheme, despite a linear growth in the number of delegations. This function is left outside of the scope of this work.

k-times anonymity was introduced for authentication, group signatures (where the group is managed by an authority that generates keys), and ring signatures (where the group is chosen *ad hoc* at the time of signing) in [TFS04], [ASY06], and [FS07b] respectively. In some schemes, the identity of the signer leaks if it produces more than k signatures. The property of k-times fully traceable anonymity [BL16] extends this concept by making it possible to trace all signatures produced *a priori* by the user that exceeds the k signature bound (and not just a pair of signatures). To the best of our knowledge, the only scheme that matches this property is the ring signature described in [BL16], and this at the cost of a signature size in $O(nk)$ where n is the number of users, and a secret key size in $O(k)$.

In [FP08], Fuchsbauer and Pointcheval mention that anonymous proxy signatures can be seen as group signatures: the delegator becomes the group manager and each delegate (*i.e.,* each group member) can sign anonymously on behalf of the manager (*i.e.,* within the group). We can therefore view our fully traceable k-times proxy signature as the first fully traceable k-times group signature. Since our aim is also to design sanitizable signatures, which have some similarities with proxy signatures (in both cases a delegator gives a delegate the power to create new signatures on its behalf), we have chosen to present our scheme as an anonymous proxy signature rather than as a group signature. In comparison with the only k-times group signature scheme [ASY06] in the literature, our scheme achieves full traceability. In return, the key/signature size is in $O(\log(k))$, whereas [ASY06] claims a constant key/signature size (note, however, that in this scheme, the delegator must produce and share a public key of size linear in k, moreover if the limit k is different for each delegate, then this key must be kept secret by each delegate, which significantly retains its practicability for large k).

Sanitizable signatures were introduced by Ateniese *et al.* [ACDMT05], who identified several security properties (unforgeability, immutability, privacy, transparency, and accountability) formally defined later in [BFF+09]. They show that privacy (the original message does not leak from the sanitized signature) is implied by transparency. Invisibility was also introduced in [ACDMT05], but received formal treatment much later in [CDK+17]. Last, but not least, unlinkability has been introduced and formalised in [BFLS10] and studied in [BPS14, FKM+16]. Only two schemes guarantee all these properties at once [BLL+19, BB21]. In this chapter, we adapt and prove all these properties on our scheme, with the exception of accountability, which consists in allowing the signer to reveal the author (*i.e.,* the original signer or the sanitizer) of a problematic signature, since this information leaks spontaneously if the sanitizer exceeds the limit of k sanitizations.

k-times traceable anonymous proxy signatures should not be confused with the k-times (not anonymous) proxy signatures introduced by Liu *et al.* in [LYMW13], where if the (non-anonymous) proxy exceeds a limit of k signatures, then its secret key leaks. This primitive is close to ours, but differs in two crucial points: (i) the proxy is not anonymous, so there is no need to trace it or link signatures, thus full traceability makes no sense in [LYMW13], and (ii) unlike [LYMW13] we do not want to leak the proxy's secret key for security reasons. Indeed, if a verifier recovers a proxy secret key, it can sign messages as a proxy without the original signer having chosen to give it this power. As a result, users which have not had access to the $k+1$ proxy signatures (including the original signer) are unaware that this verifier can impersonate the signer, which causes serious security problems in most applications. Similarly, one line of work, started in [KL06], aims to limit the sanitizer's power in various ways in sanitizable signatures [CJ10]. In particular, in [KL06, CJ10] the authors propose a scheme where if the (non-anonymous) sanitizer exceeds a limit of k signatures, then its secret key leaks, as in k-times (not anonymous) proxy signatures [LYMW13]. The differences between this primitive and ours are the same as those between k-times proxy signatures [LYMW13] and our k-times anonymous proxy signatures. Note also that our scheme guarantees more security properties simultaneously (in particular anonymity, unlinkability, and invisibility), which is much more restrictive, *e.g.* we have to ensure that the signature is fully randomizable by the sanitizer.

Finally, k-times anonymous sanitizable signatures should not be confused with γ-times sanitizable signatures [BB21], where γ bounds the number of blocks that can be modified instead of the number of times the signature can be sanitized. In the primitive introduced in [BB21], the sanitizer is not anonymous, and cannot (in the computational sense) sanitize a signature by modifying more than γ blocks. The mechanism is therefore very different, as there is no intention of triggering some secret information leak when the limit is exceeded.

4.2 Zero-knowledge Proofs as Building Blocks

Technical Summary

Our constructions require two zero-knowledge proofs for dedicated purposes. Of these two, one is called $\Pi_{<k}$ and is a range proof showing that a prover knows a committed integer η such that η is less than a specified threshold k, where η is in $\{0, \ldots, k-1\}$. Notably, our construction maintains a linear relationship between the size of the proof transcript and the number of bits of k, despite the complexity of the boolean formula thereafter.

On the other hand, a second zero-knowledge proof, called π_σ, is subsequently used to link all the tracing elements and prove their well-formedness. It uses construction bases similar to those of $\Pi_{<k}$ and is inspired by the construction of the k-times full traceable ring signature of [BL16].

We now detail the two non-interactive zero-knowledge proof constructions. The proof $\Pi_{<k}$ is described in Section 4.2.1, an example of which is given in Section 4.2.2. Subsequently, the proof π_σ is then described in Section 4.2.3.

We now detail our two constructions of non-interactive zero-knowledge proofs. The proof $\Pi_{<k}$ is described in Section 4.2.1, an example of which is given in Section 4.2.2 to provide more intuition for the reader. Next, the proof π_σ is described in Section 4.2.3.

4.2.1 The Two Zero-knowledge Proofs

The $\Pi_{<k}$ proof guarantees the limit of k signatures authorised by the delegator to its proxy. In practice, the $\Pi_{<k}$ proof ensures that the prover knows an elements s and an integer η such that (i) $\widetilde{y}_i = y_{i,j}^s$ and $\widetilde{\mathsf{pk}}_i = \mathsf{pk}_{i,j}^s$ are well formed according to s, some integer η of l bits (*i.e.,* $l = \log_2(k)$) and the public values $y_{i,j}$ and $\mathsf{ppk}_{i,j}$, for $i \in \{0, \ldots, k-1\}$ and $j \in \{0,1\}$. And (ii) $\eta < k$. Proving (i) is equivalent to prove $(\widetilde{g}_1 = g_1^s$ and $\widetilde{y}_i = y_{i,0}^s$ and $\widetilde{\mathsf{pk}}_i = \mathsf{pk}_{i,0}^s)$ or $(\widetilde{g}_1 = g_1^s$ and $\widetilde{y}_i = y_{i,1}^s$ and $\widetilde{\mathsf{pk}}_i = \mathsf{pk}_{i,1}^s)$ for all $i \in [\![0, l]\!]$. Formulated now based of the Camenisch and Stadler [CS97a] notation introduced in Definition 20 on page 42, we obtain:

$$\mathsf{ZK}\left\{ s : \bigwedge_{i=0}^{l} \bigvee_{j=0}^{1} \left(\widetilde{g}_1 = g_1^s \wedge \widetilde{y}_i = y_{i,j}^s \wedge \widetilde{\mathsf{pk}}_i = \mathsf{pk}_{i,j}^s \right) \right\}.$$

The transcript of this proof is linear in l. On the other hand, proving (ii) consists in proving $\eta < k$, where each bit $\eta[i]$ of η is committed in $\widetilde{y}_i = y_{i,\eta[i]}^s$. So to prove that η is smaller than k, we need to compare k and η as binary words across commitments $\widetilde{y}_i$, by going through the bits, from most to least significant. For instance, using $k = 1001101$, proving that $\eta < k$ consists in proving that $\eta[0] = 0$ or $(\eta[1] = 0$ and $\eta[2] = 0$ and $(\eta[3] = 0$ or $(\eta[4] = 0$ or $(\eta[5] = 0$ and $\eta[6] = 0)))$. In this case, the required proof is:

$$\mathsf{ZK}\left\{ s : \begin{array}{l} (\widetilde{g}_1 = g_1^s \wedge \widetilde{y}_0 = y_{0,0}^s) \vee ((\widetilde{g}_1 = g_1^s \wedge \widetilde{y}_1 = y_{1,0}^s) \wedge (\widetilde{g}_1 = g_1^s \wedge \widetilde{y}_2 = y_{2,0}^s) \\ \wedge((\widetilde{g}_1 = g_1^s \wedge \widetilde{y}_3 = y_{3,0}^s) \vee ((\widetilde{g}_1 = g_1^s \wedge \widetilde{y}_4 = y_{4,0}^s) \\ \vee((\widetilde{g}_1 = g_1^s \wedge \widetilde{y}_5 = y_{5,0}^s) \wedge (\widetilde{g}_1 = g_1^s \wedge \widetilde{y}_6 = y_{6,0}^s))))) \end{array} \right\}.$$

This technique can be generalised as follows. Let $(i_j)_{0 \leq j \leq n}$ be the indices of the 1's in the binary word k. Note that i_0 is always 1. Proving that $\eta < k$ consists of providing the following proof:

$$\mathsf{ZK}\left\{ s : \begin{array}{l} (\widetilde{g}_1 = g_1^s \wedge \widetilde{y}_{i_0} = y_{i_0,0}^s) \vee (\bigwedge_{i=i_0+1}^{i_1-1}(\widetilde{g}_1 = g_1^s \wedge \widetilde{y}_i = y_{i,0}^s) \wedge \\ ((\widetilde{g}_1 = g_1^s \wedge \widetilde{y}_{i_1} = y_{i_1,0}^s) \vee (\bigwedge_{i=i_1+1}^{i_2-1}(\widetilde{g}_1 = g_1^s \wedge \widetilde{y}_i = y_{i,0}^s) \wedge (\ldots \\ (\widetilde{g}_1 = g_1^s \wedge \widetilde{y}_{i_n} = y_{i_n,0}^s) \vee (\bigwedge_{i=i_n+1}^{l}(\widetilde{g}_1 = g_1^s \wedge \widetilde{y}_i = y_{i,0}^s)) \ldots)))) \end{array} \right\}$$

The relation of this proof is a boolean combination of l proofs of equality of discrete logarithms. Using the techniques presented above, we thus obtain a proof whose transcript size is l times the transcript size of a proof of equality of discrete logarithms. This may seem surprising, since the development of the boolean formula gives on the order of l^2 terms, however the generic transformations we use for the `and` and the `or` proofs depend

on how the formula is expressed, and the size of the proofs is linear in the number of termes in the formula.

Instantiating the proof $\Pi_{<k}$, requires several building blocks. In [CP93], Chaum and Pedersen introduce a zero-konwledge proof of knowledge for discrete logarithm equality $\mathsf{ZK}\{x : y_1 = g_1^x \wedge y_2 = g_2^x\}$ in a group of prime order p. This proof is a sigma protocol: the prover sends a commitment, the verifier sends a challenge (chosen in $\mathbb{Z}_p^*$), and the prover sends a response. This proof can be extended to prove the equality of more than two discret logarithms. In this case the size of the resulted transcript is linear in the number of statements. In general, if two sigma protocols for two instances ϕ_1 and ϕ_2 and two relations $\mathcal{R}_1$ and $\mathcal{R}_2$ use the same challenge space, it is possible to merge the proofs by using the same challenge in order to obtain an `and`-proof $\mathsf{ZK}\{w_1, w_2 : (w_1, \phi_1) \in \mathcal{R}_1 \wedge (w_2, \phi_2) \in \mathcal{R}_2\}$. This method can also be extended to any number of instances. In [CDS94], Cramer *et al.* proposed a zero-knowledge proof to prove the knowledge of the witness corresponding to one of two instances $\mathsf{ZK}\{w : (w, \phi_1) \in \mathcal{R}_1 \vee (w, \phi_2) \in \mathcal{R}_2\}$, under the hypothesis that $\mathsf{ZK}\{w : (w, \phi_1) \in \mathcal{R}_1\}$ and $\mathsf{ZK}\{w : (w, \phi_2) \in \mathcal{R}_2\}$ are sigma protocols that use the same challenge space. The challenge space of the resulting proof remains the same as that of the two combined proofs. The method can be extended to prove the knowledge of a witness in relation to one instance amongst n, in which case the transcript size is equal to the sum of the transcript sizes of the proofs of each instance. In Section 4.2.2, we detail an example to illustrate this point. Finally, we use the Fiat-Shamir [FS87] transform to change these proofs into non-interactive ones. The proof $\Pi_{<k}$ is the composition of the two proofs presented above.

Property 7: Security of $\Pi_{<k}$

The zero-knowledge proofs $\Pi_{<k}$ is perfectly complete, zero-knowledge, sound and simulation-extractable.

Proof. This proof is constructed based of the framework provided in [CP93], [CDS94] and the Fiat-Shamir transformation [FS87], which makes the Σ-protocol non-interactive. In particular, the security of the zero-knowledge proof for the equality of the discrete logarithm comes from the results of [CP93] (`and` statements) and the security of the `or` statements appearing in the zero-knowledge proof comes from the results of [CDS94]. □

4.2.2 An Example for the Proof $\Pi_{<k}$

In this section, we give more details about the structure of the proof $\Pi_{<k}$, then we show an example that illustrates how the proof works and why it is linear in l. We first recall some facts about Σ-protocols and `or`-proofs. A Σ-protocol is made up of three interractions, enabling the exchange of a commitment R, a challenge c, and a response z. Usually, the simulator of such a protocol for some discrete logarithm relation in a group of prime order p randomly picks a challenge $c \in \mathbb{Z}_p^*$ and a response $z \in \mathbb{Z}_p^*$, then computes the comitment R from (c, z) to complete the simulated transcript (R, c, z).

The Cramer *et al.* `or`-proof transformation [CDS94] transfoms n Σ-protocols sharing the same challenge space for the respectives statements/relations $(\phi_i)_{i\in[\![n]\!]}$ and $(\mathcal{R}_i)_{i\in[\![n]\!]}$

denoted $\mathsf{ZK}\{w : (w, \phi_i) \in \mathcal{R}_i\}$ into an `or`-proof Σ-protocol $\mathsf{ZK}\left\{w : \bigvee_{i\in[\![n]\!]}(w, \phi_i) \in \mathcal{R}_i\right\}$. This transformation works as follows: assume that the prover knows the witness w_j for the statement/relation ϕ_j and $\mathcal{R}_j$. It first produces the commitment R_j for ϕ_j as in the proof $\mathsf{ZK}\{w : (w, \phi_j) \in \mathcal{R}_j\}$, then simulates the transcripts (R_i, c_i, z_i) for the other statements ϕ_i where $i \neq j$. It sends the commitments $(R_i)_{i\in[\![n]\!]}$ to the verifier and receives the challenge c. The prover then computes $c_j = c \oplus \bigoplus_{i\in[\![n]\!];i\neq j} c_i$, computes the response z_j from w_j, R_j and c_j as in $\mathsf{ZK}\{w : (w, \phi_j) \in \mathcal{R}_j\}$, and returns $(c_i, z_i)_{i\in[\![n]\!]}$ to the verifier. The verifier checks that $c = \bigoplus_{i\in[\![n]\!]} c_i$, and that each transcript (R_i, c_i, z_i) is valid according to ϕ_i and $\mathcal{R}_i$ for $i \in [\![n]\!]$.

On the other hand, the `and`-proof transformation that we use to construct zero-knowledge proofs $\mathsf{ZK}\left\{(w_i)_{i\in[\![n]\!]} : \bigwedge_{i\in[\![n]\!]}(w, \phi_i) \in \mathcal{R}_i\right\}$ consists of executing the proofs $\mathsf{ZK}\{w_i : (w_i, \phi_i) \in \mathcal{R}_i\}$ in parallel by using a unique challenge c: the prover sends the commitments $(R_i)_{i\in[\![n]\!]}$, receives a challenge c, and outputs the responses $(z_i)_{i\in[\![n]\!]}$ such that each (R_i, c, z_i) is a valid transcript for the statement/relation $(\phi_i, \mathcal{R}_i)$.

In what follows, we will show how the second part of the proof $\Pi_{<k}$ works for the example $k = 1001101$ given in Section 4.2.1. We stress that the aim of this proof is to prove that the witness is a number η for $\eta < k$.

$$\mathsf{ZK}\left\{ s : \begin{array}{l} (\widetilde{g}_1 = \widehat{g}_1^s \wedge \widetilde{y}_0 = \widehat{y}_{0,0}^s) \vee ((\widetilde{g}_1 = \widehat{g}_1^s \wedge \widetilde{y}_1 = \widehat{y}_{1,0}^s) \\ \wedge(\widetilde{g}_1 = \widehat{g}_1^s \wedge \widetilde{y}_2 = \widehat{y}_{2,0}^s) \wedge ((\widetilde{g}_1 = \widehat{g}_1^s \wedge \widetilde{y}_3 = \widehat{y}_{3,0}^s) \\ \vee((\widetilde{g}_1 = \widehat{g}_1^s \wedge \widetilde{y}_4 = \widehat{y}_{4,0}^s) \vee ((\widetilde{g}_1 = \widehat{g}_1^s \wedge \widetilde{y}_5 = \widehat{y}_{5,0}^s) \\ \wedge(\widetilde{g}_1 = \widehat{g}_1^s \wedge \widetilde{y}_6 = \widehat{y}_{6,0}^s))))) \end{array} \right\}.$$

Throughout this section:

- $\mathcal{R}$ denotes the relation:

$$\begin{array}{l} (\widetilde{g}_1 = \widehat{g}_1^s \wedge \widetilde{y}_0 = \widehat{y}_{0,0}^s) \vee ((\widetilde{g}_1 = \widehat{g}_1^s \wedge \widetilde{y}_1 = \widehat{y}_{1,0}^s) \wedge (\widetilde{g}_1 = \widehat{g}_1^s \wedge \widetilde{y}_2 = \widehat{y}_{2,0}^s) \\ \wedge((\widetilde{g}_1 = \widehat{g}_1^s \wedge \widetilde{y}_3 = \widehat{y}_{3,0}^s) \vee ((\widetilde{g}_1 = \widehat{g}_1^s \wedge \widetilde{y}_4 = \widehat{y}_{4,0}^s) \vee ((\widetilde{g}_1 = \widehat{g}_1^s \wedge \widetilde{y}_5 = \widehat{y}_{5,0}^s) \\ \wedge(\widetilde{g}_1 = \widehat{g}_1^s \wedge \widetilde{y}_6 = \widehat{y}_{6,0}^s))))). \end{array}$$

- $\mathcal{R}_0$ denotes the relation $(\widetilde{g}_1 = \widehat{g}_1^s \wedge \widetilde{y}_0 = \widehat{y}_{0,0}^s)$.

- $\mathcal{R}_{1,2-}$ (as an abbreviation of $\mathcal{R}_{1,2,3,4,5,6}$) denotes the relation:

$$\begin{array}{l} (\widetilde{g}_1 = \widehat{g}_1^s \wedge \widetilde{y}_1 = \widehat{y}_{1,0}^s) \wedge (\widetilde{g}_1 = \widehat{g}_1^s \wedge \widetilde{y}_2 = \widehat{y}_{2,0}^s) \wedge ((\widetilde{g}_1 = \widehat{g}_1^s \wedge \widetilde{y}_3 = \widehat{y}_{3,0}^s) \\ \vee((\widetilde{g}_1 = \widehat{g}_1^s \wedge \widetilde{y}_4 = \widehat{y}_{4,0}^s) \vee ((\widetilde{g}_1 = \widehat{g}_1^s \wedge \widetilde{y}_5 = \widehat{y}_{5,0}^s) \wedge (\widetilde{g}_1 = \widehat{g}_1^s \wedge \widetilde{y}_6 = \widehat{y}_{6,0}^s)))). \end{array}$$

- $\mathcal{R}_3$ denotes the relation $(\widetilde{g}_1 = \widehat{g}_1^s \wedge \widetilde{y}_3 = \widehat{y}_{3,0}^s)$.

- $\mathcal{R}_4$ denotes the relation $(\widetilde{g}_1 = \widehat{g}_1^s \wedge \widetilde{y}_4 = \widehat{y}_{4,0}^s)$.

- $\mathcal{R}_{5,6}$ denotes the relation $(\widetilde{g}_1 = \widehat{g}_1^s \wedge \widetilde{y}_5 = \widehat{y}_{5,0}^s) \wedge (\widetilde{g}_1 = \widehat{g}_1^s \wedge \widetilde{y}_6 = \widehat{y}_{6,0}^s)$.

Moreover, (R_x, c_x, z_x) will denote the transcript for the relation $\mathcal{R}_x$ in the proof. According to the boolean structure of the relation $\mathcal{R}$, the challenges c chosen by the verifier and the challenges $c_0, c_{1,2-}, c_3, c_4$ and $c_{5,6}$ sent by the verifier must verify the following

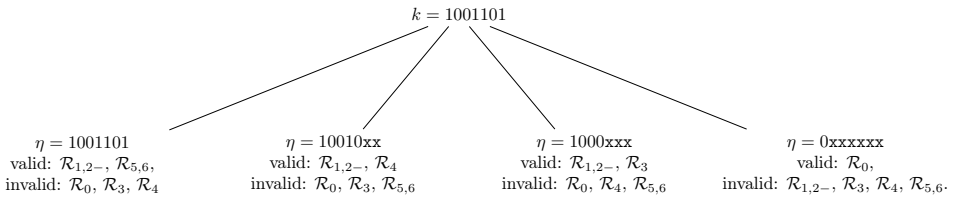


Figure 4.1: Tree Structure of the $\Pi_{<k}$ Proof for $k = 1001101$ and a Valid η.

equations:

$$c = c_0 \oplus c_{1,2-} \ ;$$
$$c_{1,2-} = c_3 \oplus c_4 \oplus c_{5,6} \ .$$

If the prover is honest (*i.e.,* $\mathcal{R}$ holds), then we have the following cases also highlighted in Figure 4.1.

Case $\eta = 1001100$: the relations $\mathcal{R}_{1,2-}$ and $\mathcal{R}_{5,6}$ hold, but the relations $\mathcal{R}_0$, $\mathcal{R}_3$ and $\mathcal{R}_4$ are not verified. The prover chooses (c_0, z_0), (c_3, z_3) and (c_4, z_4), then simulates the transcripts for these relations. It then receives c from the verifier, which fixes the values of $c_{1,2-}$ and $c_{5,6}$:

$$c_{1,2-} = c_0 \oplus c \ ;$$
$$c_{5,6} = c_3 \oplus c_4 \oplus c_{1,2-} \ .$$

Since $\mathcal{R}_{1,2-}$ and $\mathcal{R}_{5,6}$ hold, the prover is able to compute the responses $z_{1,2-}$ and $z_{5,6}$ from $c_{1,2-}$ and $c_{5,6}$.

Case $\eta = 10010\mathtt{xx}$ (where each $\mathtt{x}$ can be replaced by any bit): the relations $\mathcal{R}_{1,2-}$ and $\mathcal{R}_4$ hold, but the relations $\mathcal{R}_0$, $\mathcal{R}_3$ and $\mathcal{R}_{5,6}$ may not be verified. The prover chooses (c_0, z_0), (c_3, z_3) and $(c_{5,6}, z_{5,6})$, then simulates the transcripts for these relations. It then receives c from the verifier, which fixes the values of $c_{1,2-}$ and c_4:

$$c_{1,2-} = c_0 \oplus c \ ;$$
$$c_4 = c_3 \oplus c_{5,6} \oplus c_{1,2-} \ .$$

Since $\mathcal{R}_{1,2-}$ and $\mathcal{R}_4$ hold, the prover is able to compute the responses $z_{1,2-}$ and z_4 from $c_{1,2-}$ and c_4.

Case $\eta = 1000\mathtt{xxx}$ (where each $\mathtt{x}$ can be replaced by any bit): the relations $\mathcal{R}_{1,2-}$ and $\mathcal{R}_3$ hold, but the relations $\mathcal{R}_0$, $\mathcal{R}_4$ and $\mathcal{R}_{5,6}$ may not be verified. The prover chooses (c_0, z_0), (c_4, z_4) and $(c_{5,6}, z_{5,6})$, then simulates the transcripts for these relations. It receives c from the verifier, which fixes the values of $c_{1,2-}$ and c_3:

$$c_{1,2-} = c_0 \oplus c \ ;$$
$$c_3 = c_4 \oplus c_{5,6} \oplus c_{1,2-} \ .$$

Since $\mathcal{R}_{1,2-}$ and $\mathcal{R}_3$ hold, the prover is able to compute the responses $z_{1,2-}$ and z_3 from $c_{1,2-}$ and c_3.

Case $\eta = 0$xxxxxx (where each x can be replaced by any bit): the relation $\mathcal{R}_0$ holds, but the relations $\mathcal{R}_{1,2-}$, $\mathcal{R}_3$, $\mathcal{R}_4$ and $\mathcal{R}_{5,6}$ may not be verified. The prover chooses $z_{1,2-}$, (c_3, z_3), (c_4, z_4) and $(c_{5,6}, z_{5,6})$, which fixes the value $c_{1,2-}$:

$$c_{1,2-} = c_3 \oplus c_4 \oplus c_{5,6} \ .$$

The prover then simulates the transcripts for these relations and receives c from the verifier, which fixes the values of c_0:

$$c = c_0 \oplus c_{1,2-} \ .$$

Since $\mathcal{R}_0$ holds, the prover is able to compute the responses z_0 from c_0.

On the other hand, if the prover is dishonest (*i.e.,* $\mathcal{R}$ does not hold), then we have the following cases:

Case $\eta = 1001101$: the relations $\mathcal{R}_0$, $\mathcal{R}_3$, $\mathcal{R}_4$, and $\mathcal{R}_{5,6}$ are not verified. The prover chooses (c_0, z_0), (c_3, z_3), (c_4, z_4), and $(c_{5,6}, z_{5,6})$ then simulates the transcripts for these relations. It then receives c from the verifier, which fixes the value of $c_{1,2-}$ in order that the equation $c_{1,2-} = c_0 \oplus c$ holds. However, since $c_{1,2-}$, c_3, c_4, and $c_{5,6}$ are fixed, the probability that the equation $c_{1,2-} = c_3 \oplus c_4 \oplus c_{5,6}$ holds is $1/p$ (each challenge is chosen in $\mathbb{Z}_p^*$), which is negligible.

Case $\eta = 100111$x (where x can be replaced by any bit): this case is similar to the previous one.

Case $\eta = 101$xxxx (where each x can be replaced by any bit): the relations $\mathcal{R}_0$ and $\mathcal{R}_{1,2-}$ are not verified. The prover chooses (c_0, z_0) and $(c_{1,2-}, z_{1,2-})$ then simulates the transcripts for these relations. It then receives c from the verifier; however the probability that the equation $c_{1,2-} = c_0 \oplus c$ holds is $1/p$, which is negligible.

Case $\eta = 11$xxxxx (where each x can be replaced by any bit): this case is similar to the previous one.

This example covers all the cases in the structure of the binary word k, and can easily be generalised. Note that the size of the transcript of this proof is linear in l. As the prover/verifier needs to check the equations on the challenges that follow a tree structure, the time complexity is quadratic in l. However, we note that the number of exponentiations remains linear in l, making this proof efficient.

4.2.3 Instantiation of the Proof π_σ

In this section, we show how to instantiate the proof π_σ later used in our k-times anonymous proxy and sanitizable signatures in Sections 4.3 and 4.4. For the sake of clarity, we write this proof with generic notations (where for any integer i, each g_i, γ_i, and h_i are elements of a groups $\mathbb{G}_i$ of the same prime order p), later specialised in our

signatures. The proof π_σ is as follows:

$$\pi_\sigma \leftarrow \mathsf{ZK}\left\{ x, y, z: \begin{array}{l} h_1 = {g_1}^x \wedge h_2 = {g_2}^y \wedge h_3 = {g_3}^x \wedge h_4 = {g_4}^z \\ h_5 = {g_5}^x \cdot \gamma_5^y \wedge h_6 = {g_6}^x \cdot \gamma_6^y \wedge h_7 = {g_7}^y \end{array} \right\}.$$

Our construction follows the Schnorr protocol structure:

- The prover picks $(r, s, t) \xleftarrow{\$} \mathbb{Z}_p^3$ and sets $R_1 = g_1^r$; $S_2 = g_2^s$; $R_3 = g_3^r$; $T_4 = g_4^t$; $R_5 = g_5^r$; $S_5 = g_5^s$; $R_6 = g_6^s$; $S_6 = g_6^s$; and $S_7 = g_7^s$. The prover sends $(R_1, S_2, R_3, T_4, R_5, S_5, R_6, S_5, S_7)$ to the verifier.
- The verifier picks a challenge $c \xleftarrow{\$} \mathbb{Z}_p$ and sends it to the prover.
- The prover computes $\alpha = r + x \cdot c$, $\beta = s + y \cdot c$, and $\delta = t + z \cdot c$, then sends (α, β, γ) to the verifier.
- If the following equations holds, then the verifier accepts the proof: $g_1^\alpha = R_1 \cdot h_1^c$; $g_2^\beta = S_2 \cdot h_2^c$; $g_3^\alpha = R_3 \cdot h_3^c$; $g_4^\delta = T_4 \cdot h_4^c$; $g_5^\alpha \cdot \gamma_5^\beta = R_5 \cdot S_5 \cdot h_5^c$; $g_6^\alpha \cdot \gamma_6^\beta = R_6 \cdot S_6 \cdot h_6^c$; and $g_7^\beta = S_7 \cdot h_7^c$. Else the verifier rejects.

This proof is **complete** by construction.

To show that this proof is **sound**, we show that knowing two valid transcripts $\tau_0 = ((R_1, S_2, R_3, T_4, R_5, S_5, R_6, S_5, S_7), c_0, (\alpha_0, \beta_0, \gamma_0))$ and $\tau_1 = ((R_1, S_2, R_3, T_4, R_5, S_5, R_6, S_5, S_7), c_1, (\alpha_1, \beta_1, \gamma_1))$ both using the same commitment $(R_1, S_2, R_3, T_4, R_5, S_5, R_6, S_5, S_7)$ but different challenges c_0 and c_1, it is possible to deduce (x, y, z) in polynomial-time (special soundness). Since the two transcripts are valid, we have: $g_1^{\alpha_0} = R_1 \cdot h_1^{c_0}$; $g_2^{\beta_0} = S_2 \cdot h_2^{c_0}$; $g_3^{\alpha_0} = R_3 \cdot h_3^{c_0}$; $g_4^{\delta_0} = T_4 \cdot h_4^{c_0}$; $g_5^{\alpha_0} \cdot \gamma_5^{\beta_0} = R_5 \cdot S_5 \cdot h_5^{c_0}$; $g_6^{\alpha_0} \cdot \gamma_6^{\beta_0} = R_6 \cdot S_6 \cdot h_6^{c_0}$; $g_7^{\beta_0} = S_7 \cdot h_7^{c_0}$; $g_1^{\alpha_1} = R_1 \cdot h_1^{c_1}$; $g_2^{\beta_1} = S_2 \cdot h_2^{c_1}$; $g_3^{\alpha_1} = R_3 \cdot h_3^{c_1}$; $g_4^{\delta_1} = T_4 \cdot h_4^{c_1}$; $g_5^{\alpha_1} \cdot \gamma_5^{\beta_1} = R_5 \cdot S_5 \cdot h_5^{c_1}$; $g_6^{\alpha_1} \cdot \gamma_6^{\beta_1} = R_6 \cdot S_6 \cdot h_6^{c_1}$; and $g_7^{\beta_1} = S_7 \cdot h_7^{c_1}$. Setting $x = (\alpha_1 - \alpha_0)/(c_1 - c_0)$; $y = (\beta_1 - \beta_0)/(c_1 - c_0)$; and $z = (\delta_1 - \delta_0)/(c_1 - c_0)$ and using the equations above, we find that: $h_1 = {g_1}^x$; $h_2 = {g_2}^y$; $h_3 = {g_3}^x$: $h_4 = {g_4}^z$; $h_5 = {g_5}^x \cdot \gamma_5^y$; $h_6 = {g_6}^x \cdot \gamma_6^y$; and $h_7 = {g_7}^y$, which concludes the proof of soundness.

We show that this proof is **zero-knowledge** by giving a polynomial-time simulator that outputs transcrpits indistinguishable from the transcripts of the real protocol without using the secret value (x, y, z). The simulator picks $(c, \alpha, \beta, \delta) \xleftarrow{\$} \mathbb{Z}_p^4$; $R_5 \xleftarrow{\$} \mathbb{G}_5$; and $R_6 \xleftarrow{\$} \mathbb{G}_6$. Then the simulator computes: $R_1 = g_1^\alpha / h_1^c$; $S_2 = g_2^\beta / h_2^c$; $R_3 = g_3^\alpha / h_3^c$; $T_4 = g_4^\delta / h_4^c$; $S_5 = (g_5^\alpha \cdot \gamma_5^\beta)/(R_5 \cdot h_5^c)$; $S_6 = (g_6^\alpha \cdot \gamma_6^\beta)/(R_6 \cdot h_6^c)$; and $S_7 = g_7^\beta / h_7^c$. The simulator returns $((R_1, S_2, R_3, T_4, R_5, S_5, R_6, S_5, S_7), c, (\alpha, \beta, \gamma))$.

Finally, as this proof is a Σ-protocol, it can be made **non-interactive** using the Fiat-Shamir transformation [FS87].

4.3 k-Times Anonymous Proxy Signatures

Technical Summary

Anonymous proxy signatures schemes [FP08] allow a designated proxy to sign messages on behalf of an original signer, ensuring the signer's anonymity. In this section,

we bound the delegation provided to the proxy in order to limit it to a maximum of k signatures and guarantee fully traceability by the verifier in the event of over-passed limit. This new primitive is called k-times full traceable proxy signatures. For simplicity, we use the term *k-Times Anonymous Proxy Signatures*, provide a security model for this new primitive and instantiate it based on an SPS scheme (Definition 16 on page 36) and the NIZK proofs presented in Section 4.2. We also provide a security proof to demonstrate the security of our scheme.

k-Times Anonymous Proxy Signatures involves three main steps: (1) *key generation*: both the original signer and proxy generate key pairs $(\mathsf{sk}, \mathsf{pk})$ and $(\mathsf{psk}, \mathsf{ppk})$, respectively; (2) *delegation*: the original signer generates a delegation token del, enabling the proxy to sign messages on its behalf. (3) *signing and verification*: the proxy signs a message m using del and psk to produce a signature σ. The verifier checks the signature's validity using pk without learning the proxy signer's identity denoted as ppk, thereby preserving its anonymity.

4.3.1 Security Model for k-Times Anonymous Proxy Signatures

We start our discussion on k-times (fully traceable) anonymous proxy signatures by giving a formal definition and security model. In this primitive, a signer can delegate to a proxy the authority to anonymously produce at most k signatures. To do this, the signer generates a delegation certificate (denoted cert) via the algorithm $\mathsf{Delegate}_{\mathsf{k\text{-}APS}}$, using the proxy's public key and the limit k as input. To produce a proxy signature, the proxy uses this delegation with an integer $\eta \in \{0, \ldots, k-1\}$ that must be different for each of the k signatures. Note that η must not appear in the signature to preserve anonymity (we will describe the corresponding security model in more detail later in this section), so it is not given as input to the verification algorithm. If the proxy decides to produce more than k signatures, it will be forced to use the same index η twice, triggering a mechanism that allows any user to link these two signatures using an algorithm $\mathsf{Link}_{\mathsf{k\text{-}APS}}$, and to extract the identity of the proxy. The algorithm $\mathsf{Link}_{\mathsf{k\text{-}APS}}$ also returns a token w which, when used with a signature as input to the $\mathsf{Trace}_{\mathsf{k\text{-}APS}}$ algorithm, indicates whether or not the signature was generated by the same proxy, enabeling the traceability of all the signatures generated by the proxy in the past. Note that the signer can extend the limit by generating new delegations for the same proxy.

Definition 27: k-times Anonymous Proxy Signature scheme - k-APS

A *k-times Anonymous Proxy Signature scheme* (k-APS) is a tuple of algorithms:

$\mathsf{Setup}_{\mathsf{k\text{-}APS}}(1^\lambda)$: given a security parameter, returns a public parameter pp. Note that pp is assumed to be an implicit input to all the following algorithms.

$\mathsf{Gen}_{\mathsf{k\text{-}APS}}(1^\lambda, k)$: given a limit $k \in \mathbb{N}$, returns the signer's secret/public keys $(\mathsf{sk}, \mathsf{pk})$.

$\mathsf{PKeyGen}_{\mathsf{k\text{-}APS}}(1^\lambda)$: returns the proxy's secret/public keys $(\mathsf{psk}, \mathsf{ppk})$.

$\mathsf{Delegate}_{\mathsf{k\text{-}APS}}(\mathsf{sk}, \mathsf{ppk}, l)$: given the keys sk, ppk and $l \leq k$, returns a delegation certificate cert.

$\mathsf{Sign}_{\mathsf{k\text{-}APS}}(\mathsf{pk}, \mathsf{psk}, m, \mathsf{cert}, \eta)$: given the keys pk, psk, a message m, a certificate cert, and an index η, returns a signature σ.

$\mathsf{Verif}_{\mathsf{k\text{-}APS}}(\mathsf{pk}, m, \sigma)$: given the key pk, a message m, and a signature σ, returns 0 or 1 (for *reject* or *accept*).

$\mathsf{Link}_{\mathsf{k\text{-}APS}}(\mathsf{pk}, m, \sigma, m', \sigma')$: given a key pk and two pairs (m, σ), (m', σ'), returns an identity ppk and a witness w or $\perp$ in case of failure.

$\mathsf{Trace}_{\mathsf{k\text{-}APS}}(w, \sigma)$: given a witness w and a signature σ, returns 0 or 1.

We require that k-APS meets *Correctness*, *Unforgeability*, *Anonymity*, *Traceability* and *Non-frameability* as defined thereafter.

A k-APS is said to be *correct* if, using keys and certificates honestly generated by the algorithms $\mathsf{Gen}_{\mathsf{k\text{-}APS}}$, $\mathsf{PKeyGen}_{\mathsf{k\text{-}APS}}$, and $\mathsf{Delegate}_{\mathsf{k\text{-}APS}}$, (i) any signature produced by the algorithm $\mathsf{Sign}_{\mathsf{k\text{-}APS}}$ is verified by the algorithm $\mathsf{Verif}_{\mathsf{k\text{-}APS}}$ using the signer public key, (ii) 2 signatures are linked by the algorithm $\mathsf{Link}_{\mathsf{k\text{-}APS}}$ which outputs the corresponding public key if and only if they were produced with the same delegation certificate and the same η, and (iii) the algorithm $\mathsf{Trace}_{\mathsf{k\text{-}APS}}$ returns 1 on the token outputted by $\mathsf{Link}_{\mathsf{k\text{-}APS}}$ and any of the signatures produced from this delegation certificate.

Our security model is inspired both by that of anonymous proxy signatures [FP08] and that of k-times full traceability for ring signatures [BL16].

Before looking at each of the experiments, we highlight the type of oracle on which they rely. These oracles are: a *registration oracle*, a *delegation oracle*, a *signature oracle* and also in some experiments a *challenge oracle*. These oracles are then specialised to model each of the properties.

$\mathcal{O}_{\mathsf{Register}}$. The *registration oracle* allows the adversary to request the registration of an entity, requesting the challenger to generate an entity with new keys based on the execution of the $\mathsf{PKeyGen}_{\mathsf{k\text{-}APS}}$ algorithm.

$\mathcal{O}_{\mathsf{Delegate}}$. The *delegation oracle* allows the adversary to request the delegation to a specified entity based on the execution of the $\mathsf{Delegate}_{\mathsf{k\text{-}APS}}$ algorithm.

$\mathcal{O}_{\mathsf{Sign}}$. The *signing oracle* allows the adversary to request a signature from a designated registered entity which has obtained a delegation. It is produced based on the execution of the $\mathsf{Sign}_{\mathsf{k\text{-}APS}}$ algorithm.

$\mathcal{O}_{\mathsf{chal}}$. The *challenge oracle* allows the adversary to request chalenges, here signatures from an unknown signer produced by running the $\mathsf{Sign}_{\mathsf{k\text{-}APS}}$ algorithm. This oracle ressembles, in its purpose, the $\mathcal{LoR}$ oracle of Chapter 3.

For each of the oracles, which have yet to be defined, the underlined entries correspond to those chosen by the adversary, the other inputs being provided by the challenger. Experiments use multisets (sets that may contain the same element multiple times) that are considered to be global variables (and can therefore be accessed and modified in oracles): $\mathcal{U}$ stores the registered users, $\mathcal{D}$ stores the delegations, $\mathcal{S}$ stores

the produced signatures, and $\mathcal{H}$ stores the signature indexes. The main reason we instantiate these oracles independently for each experiment is that the sets need to be populated with elements that differ according to each property.

Aside from the fundamental requirement of *correctness*, the security principles of k-APS encompass several crucial aspects: *Unforgeability*, *Anonymity*, *Traceability*, and *Non-frameability*. In Figure 4.2, we start by introducing an oracle common to the experiments of anonymity and non-frameability.

Oracle $\mathcal{O}_{\mathsf{Delegate}}(\mathsf{sk}, \underline{\mathsf{ppk}, l \leq k})$

$\mathsf{cert} \leftarrow \mathsf{Delegate}_{\mathsf{k\text{-}APS}}(\mathsf{sk}, \mathsf{ppk}, l)$
$\mathcal{D} \leftarrow \mathcal{D} \sqcup \{(\mathsf{ppk}, \mathsf{cert}, l)\}$
return cert

Figure 4.2: Description of the $\mathcal{O}_{\mathsf{Delegate}}$ Oracle for k-APS.

Unforgeability. This property ensures that an adversary playing the role of proxies will not be able to produce a signature unless they have received a delegation certificate. For this property to hold, a PPT adversary $\mathcal{A}$ must forge a valid fresh message/signature pair (m^*, σ^*) for a message that was never queried to the signature oracle.

$\mathcal{O}^{\mathsf{unf}}_{\mathsf{Register}}(\perp)$

$(\mathsf{ppk}, \mathsf{psk}) \leftarrow \mathsf{PKeyGen}_{\mathsf{k\text{-}APS}}(1^\lambda)$
$\mathcal{U} \leftarrow \mathcal{U} \cup \{(\mathsf{ppk}, \mathsf{psk})\}$
return ppk

$\mathcal{O}^{\mathsf{unf}}_{\mathsf{Delegate}}(\mathsf{sk}, \underline{\mathsf{ppk}, l \leq k})$

if $\exists \mathsf{psk}, \text{s.t.} (\mathsf{ppk}, \mathsf{psk}) \in \mathcal{U}$,
  $\mathsf{cert} \leftarrow \mathsf{Delegate}_{\mathsf{k\text{-}APS}}(\mathsf{sk}, \mathsf{ppk}, l)$
  $\mathcal{D} \leftarrow \mathcal{D} \sqcup \{(\mathsf{ppk}, \mathsf{cert}, l)\}$
  return cert
else return $\perp$

$\mathcal{O}^{\mathsf{unf}}_{\mathsf{Sign}}(\mathsf{pk}, \underline{\mathsf{ppk}, \mathsf{cert}, \eta, m})$

if $\exists \mathsf{psk}, \text{s.t.} (\mathsf{ppk}, \mathsf{psk}) \in \mathcal{U}$,
  $\sigma \leftarrow \mathsf{Sign}_{\mathsf{k\text{-}APS}}(\mathsf{pk}, \mathsf{psk}, m, \mathsf{cert}, \eta)$
  $\mathcal{S} \leftarrow \mathcal{S} \cup \{(m, \sigma)\}$
  return σ
else return $\perp$

(a) Description of the $\mathcal{O}^{\mathsf{unf}}_{\mathsf{Register}}$, $\mathcal{O}^{\mathsf{unf}}_{\mathsf{Delegate}}$ and $\mathcal{O}^{\mathsf{unf}}_{\mathsf{Sign}}$ Oracles.

$\mathsf{Exp}^{\mathsf{unf}}_{\mathcal{A},\mathsf{k\text{-}APS}}(1^\lambda)$

$\mathcal{D}, \mathcal{S} \leftarrow \emptyset$
$\mathsf{pp} \leftarrow \mathsf{Setup}_{\mathsf{k\text{-}APS}}(1^\lambda)$
$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}_{\mathsf{k\text{-}APS}}(1^\lambda, k)$
$(m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathcal{O}^{\mathsf{unf}}_{\mathsf{Register}}, \mathcal{O}^{\mathsf{unf}}_{\mathsf{Delegate}}, \mathcal{O}^{\mathsf{unf}}_{\mathsf{Sign}}}(\mathsf{pk}, k)$
return $\mathsf{Verif}_{\mathsf{k\text{-}APS}}(\mathsf{pk}, m^*, \sigma^*) \wedge ((m^*, \cdot) \notin \mathcal{S})$

(b) Description of Unforgeability Experiment.

Figure 4.3: Description of Unforgeability Experiment and Oracles for k-APS.

In the unforgeability experiment, the adversary can request delegation certificates generated for non-corrupted proxies and signatures associated with their public keys through three oracles: $\mathcal{O}^{\mathsf{unf}}_{\mathsf{Register}}$, $\mathcal{O}^{\mathsf{unf}}_{\mathsf{Delegate}}$, $\mathcal{O}^{\mathsf{unf}}_{\mathsf{Sign}}$, all introduced in Figure 4.3a on the preceding page.

A k-APS is *unforgeable* if for any PPT algorithm $\mathcal{A}$, the probability $\mathsf{Adv}^{\mathsf{unf}}_{\mathcal{A},\text{k-APS}}(1^\lambda) = \Pr[\mathsf{Exp}^{\mathsf{unf}}_{\mathcal{A},\text{k-APS}}(1^\lambda) = 1]$ is negligible. $\mathsf{Exp}^{\mathsf{unf}}_{\mathcal{A},\text{k-APS}}(1^\lambda)$ is described in Figure 4.3b on the previous page.

Anonymity. Anonymity ensures that signatures do not disclose the identity of the proxy signer (corresponds to its public key) and that signatures generated by the same proxy signer remain unlinkable. Note that in our model, anonymity is only considered against concerns the signature verifiers, and not the delegator; indeed, in our application, there is no reason why the delegator should not know the identity of the proxy signing on its behalf. In fact, it is even rather preferable that the delegator is aware of the proxies allowed to generate signatures, for accountability reasons. In the corresponding experiment, the adversary chooses a limit $t \leq k$, then tries to distinguish the origin of a challenge signature produced by one of two honest proxies.

$\mathcal{O}^{\mathsf{ano}}_{\mathsf{Sign}}(t, (\mathsf{psk}_i, \mathsf{cert}_i)_{i\in\{0,1\}}, \underline{j, m})$

1 : **if** $j = b \wedge \eta_j = t - \gamma$,
2 : **return** $\perp$
3 : **if** $j = 1 - b \wedge \eta_j = t - 1$,
4 : **return** $\perp$
5 : $\sigma \leftarrow \mathsf{Sign}_{\text{k-APS}}(\mathsf{pk}, \mathsf{psk}_j, m, \mathsf{cert}_j, \eta_j)$
6 : $\eta_j \leftarrow \eta_j + 1$
7 : **return** σ

Oracle $\mathcal{O}^{\mathsf{ano}}_{\mathsf{chal}}(t, (\mathsf{psk}_i, \mathsf{cert}_i)_{i\in\{0,1\}}, \underline{m})$

1 : **if** $\gamma = 0$, **return** $\perp$
2 : $\gamma \leftarrow 0$
3 : $\sigma \leftarrow \mathcal{O}^{\mathsf{ano}}_{\mathsf{Sign}}(b, t, (\mathsf{psk}_i, \mathsf{cert}_i)_{i\in\{0,1\}}, b, m)$
4 : **return** σ

(a) Description of the $\mathcal{O}^{\mathsf{ano}}_{\mathsf{Sign}}$ and $\mathcal{O}^{\mathsf{ano}}_{\mathsf{chal}}$ Oracles.

$\mathsf{Exp}^{\mathsf{ano}}_{\mathcal{A},\text{k-APS}}(1^\lambda)$

1 : $b \xleftarrow{\$} \{0,1\}, \mathcal{D} \leftarrow \emptyset, \eta_0, \eta_1 \leftarrow 0, \gamma \leftarrow 1$ // defined as global variables
2 : $\mathsf{pp} \leftarrow \mathsf{Setup}_{\text{k-APS}}(1^\lambda)$
3 : $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}_{\text{k-APS}}(1^\lambda, k)$
4 : **for** $j \in \{0,1\}$,
5 : $(\mathsf{ppk}_j, \mathsf{psk}_j) \leftarrow \mathsf{PKeyGen}_{\text{k-APS}}(1^\lambda)$
6 : $t \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{Delegate}}}(\mathsf{pk}, \mathsf{ppk}_0, \mathsf{ppk}_1)$
7 : **if** $t \notin [\![k]\!]$,
8 : **return** b
9 : **for** $j \in \{0,1\}$,
10 : $\mathsf{cert}_j \leftarrow \mathsf{Delegate}_{\text{k-APS}}(\mathsf{sk}, \mathsf{ppk}_j, t)$
11 : $b^* \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{Delegate}}, \mathcal{O}^{\mathsf{ano}}_{\mathsf{Sign}}, \mathcal{O}^{\mathsf{ano}}_{\mathsf{chal}}}(\mathsf{pk}, \mathsf{ppk}_0, \mathsf{ppk}_1)$
12 : **return** $b^* = b$

(b) Description of Anonymity Experiment.

Figure 4.4: Description of Anonymity Experiment and Oracles for k-APS.

In the anonymity experiment, the adversary can request to the oracles a maximum of $t-1$ signatures for each of the proxies, and a single signature for one of the two proxies (the one chosen by the challenger), which guarantees that the adversary cannot obtain more than t signatures for one of the two proxiesif it did then it would trivially link these signatures to the challenge, which is an inherent property of our primitive. For each of the two proxies, the signature oracle increments the index η at each signature, which ensures that the same value η is not used twice for the same proxy. Our model therefore considers adversaries trying to link two signatures from two different proxies with the same η, which would allow the adversary to infer that two signatures are not from the same proxy, and thus generally ensures that η is not leaked from the signature. The adversary can request delegation using the common oracle $\mathcal{O}_{\mathsf{Delegate}}$ provided above, signatures and the challenges of the experiment through their respectively oracles $\mathcal{O}^{\mathsf{ano}}_{\mathsf{Sign}}$ and $\mathcal{O}^{\mathsf{ano}}_{\mathsf{chal}}$ both introduced in Figure 4.4a on the facing page.

A k-APS is said to be *anonymous* if for any PPT $\mathcal{A}$, the probability $\mathsf{Adv}^{\mathsf{ano}}_{\mathcal{A},\text{k-APS}}(1^\lambda) = |\Pr[\mathsf{Exp}^{\mathsf{ano}}_{\mathcal{A},\text{k-APS}}(1^\lambda)=1]-1/2|$ is negligible. $\mathsf{Exp}^{\mathsf{ano}}_{\mathcal{A},\text{k-APS}}(1^\lambda)$ is described in Figure 4.4b on the preceding page.

$\mathsf{CheckTrace}(\mathsf{pk}, (m_i^*, \sigma_i^*)_{i=1}^{q_s})$

1 : **if** $\mathcal{D}$ is defined, $\mathcal{D} \xrightarrow{p} (\mathsf{pk}_i, \mathsf{del}_i, k_i)_{i=1}^n$

// For proxy signatures only.

2 : **if** $\mathcal{S}$ is defined $\wedge\, \exists i \in [\![q_s]\!], (m_i^*, \sigma_i^*, *, *) \in \mathcal{S}$, **return** 0

// For sanitizable signatures only.

3 : $T \leftarrow 0, \mathsf{W}, \mathsf{ID} \leftarrow \emptyset, \mathsf{diff} \leftarrow q_s - \sum_{1\le i\le n} k_i$ // diff is required to be strictly positive.

4 : **if** $(\exists i \in [\![q_s]\!], \mathsf{Verif}(\mathsf{pk}, m_i^*, \sigma_i^*) = 0) \vee (\exists i, j \in [\![q_s]\!], j \neq i, (m_i^*, \sigma_i^*) = (m_j^*, \sigma_j^*))$

5 : $\quad \vee\, (\mathsf{diff} \le 0)$, **return** 0

6 : **for** $1 \le i < j \le q_s$,

7 : $\quad (\mathsf{id}_{i,j}, w_{i,j}) \leftarrow \mathsf{Link}(\mathsf{pk}, m_i^*, \sigma_i^*, m_j^*, \sigma_j^*)$

// Try linking any two signatures.

8 : $\quad$ **if** $(\mathsf{id}_{i,j}, w_{i,j}) \neq \perp \wedge \mathsf{id}_{i,j} \notin \mathsf{ID}$,

// Identities for which signatures have been linked.

9 : $\quad\quad \mathsf{ID} \leftarrow \mathsf{ID} \cup \{\mathsf{id}_{i,j}\}, W[\mathsf{id}_{i,j}] \leftarrow W[\mathsf{id}_{i,j}] \cup \{w_{i,j}\}$

10 : $T \leftarrow \sum_{\mathsf{id}\in\mathsf{ID}} \left(\sum_{w\in\mathsf{W}[\mathsf{id}]} \left(\sum_{i=1}^{q_s} \mathsf{Trace}(w, \sigma_i) \right) \right)$

// The number of traced signatures is compared to the number of signatures allowed.

11 : **if** $T < \sum_{\mathsf{pk}_i\in\mathsf{ID}} k_i + \mathsf{diff}$, **return** 1

12 : **else** , **return** 0

Figure 4.5: CheckTrace Subroutine for the Traceability Experiment for k-APS and k-SAN.
(Algorithms Verif, Link and Trace are those of a k-times anonymous Proxy or Sanitizable Signature.)

Traceability. This property guarantees that the Trace algorithm leaks the identity of any adversary overpassing the delegation limit. In the corresponding experiment, the adversary's target is to produce more signatures than allowed by the delegator, without the $\mathsf{Link}_{\mathsf{k\text{-}APS}}$ and $\mathsf{Trace}_{\mathsf{k\text{-}APS}}$ algorithms being able to correctly link or trace the signatures. For that, the adversary can obtain multiple delegation certificates for different limits and different public keys ppk. Since each delegation certificate allows it to produce k_i signatures, it is required to produce strictly more than $\sum_{i=1}^{n} k_i$ valid signatures. The adversary wins the experiment if the number of traced signatures is less than the number of signatures that would have been traced if they had been generated honestly. This test, which is described in Figure 4.5 on the preceding page, has to take into account all the delegations that were exceeded in any execution scenario. First, note that if the limit of a delegation certificate for a public key is exceeded, then it must be possible to trace all the signatures generated by the owner of that public key, even if they were generated using a different delegation certificate. Thus, the number of signatures traced T should be at least the sum of the limits k_i of each delegation produced for each public key traced (expressed as $\sum_{\mathsf{pk}_i \in \mathsf{ID}} k_i$ in Figure 4.5 on the previous page), to which we add the number of signatures that exceed the global sum of the limits for all delegation certificates used by the adversary (expressed as $\mathsf{diff} = q_s - \sum_{i=1}^{n} k_i$ in Figure 4.5 on the preceding page, where q_s is the number of signatures output by the adversary). The adversary can request delegations using the oracle $\mathcal{O}_{\mathsf{Delegate}}$ provided above.

A $\mathsf{k\text{-}APS}$ is *traceable* if for any PPT algorithm $\mathcal{A}$, the probability $\mathsf{Adv}^{\mathsf{Trace}}_{\mathcal{A},\mathsf{k\text{-}APS}}(1^\lambda) = \Pr[\mathsf{Exp}^{\mathsf{Trace}}_{\mathcal{A},\mathsf{k\text{-}APS}}(1^\lambda) = 1]$ is negligible. $\mathsf{Exp}^{\mathsf{Trace}}_{\mathcal{A},\mathsf{k\text{-}APS}}(1^\lambda)$ is described in Figure 4.6.

$\mathsf{Exp}^{\mathsf{Trace}}_{\mathcal{A},\mathsf{k\text{-}APS}}(1^\lambda)$

1 : $\mathcal{D} \leftarrow \emptyset$
2 : $\mathsf{pp} \leftarrow \mathsf{Setup}_{\mathsf{k\text{-}APS}}(1^\lambda)$
3 : $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}_{\mathsf{k\text{-}APS}}(1^\lambda, k)$
4 : $(m_i^*, \sigma_i^*)_{i=1}^{q_s} \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{Delegate}}}(\mathsf{pk})$
5 : $b \leftarrow \mathsf{CheckTrace}(\mathsf{pk}, (m_i^*, \sigma_i^*)_{i=1}^{q_s})$
6 : **return** b

Figure 4.6: Description of the Traceability Experiment for $\mathsf{k\text{-}APS}$.

Non-frameability. This property prevents a PPT adversary from framing another user by generating malformed, yet valid, signatures. More precisely, the goal of the adversary is to output two signatures traceable to a registered proxy who remains honest. The adversary has access to oracles that can be used to register users, obtain delegation certificates, and obtain signatures for honest users. The adversary will not be able to abuse the signature oracle by producing more than k signatures for the proxy it wishes to trace. Note that in our model we implicitly assume that linking two signatures and tracing a user are performed by the same entity (we do not consider the case where an adversary only generates a tracing token w that traces an honest user without the linked signatures). In practice, this means that to delegate tracing, the delegate must

$\mathcal{O}^{\mathsf{no\text{-}Frame}}_{\mathsf{Register}}(\mathcal{U}, \underline{\mathsf{ppk}})$

1 : **if** $\mathsf{ppk} = \perp$,
2 : $\quad(\mathsf{ppk}, \mathsf{psk}) \leftarrow \mathsf{PKeyGen}_{\mathsf{k\text{-}APS}}(1^\lambda)$
3 : $\quad\mathcal{U} \leftarrow \mathcal{U} \cup \{(\mathsf{ppk}, \mathsf{psk}, |\mathcal{U}|, 1)\}$
4 : **else** $\mathcal{U} \leftarrow \mathcal{U} \cup \{(\mathsf{ppk}, \perp, |\mathcal{U}|, 0)\}$
5 : **return** ppk

$\mathcal{O}^{\mathsf{no\text{-}Frame}}_{\mathsf{Sign}}(\mathsf{pk}, (\mathsf{psk}_i, \mathsf{cert}_i)_{i\in\{0,1\}}, \underline{j, \eta, m})$

1 : **if** $\eta \in \mathcal{S}[\mathsf{ppk}]$, **return** $\perp$
2 : **if** $\exists \mathsf{psk}, i$ s.t. $(\mathsf{ppk}, \mathsf{psk}, i, 1) \in \mathcal{U}$,
3 : $\quad\mathcal{S}[\mathsf{ppk}] \leftarrow \mathcal{S}[\mathsf{ppk}] \cup \{\eta\}$
4 : $\quad$**return** $\mathsf{Sign}_{\mathsf{k\text{-}APS}}(\mathsf{pk}, \mathsf{psk}_j, m, \mathsf{cert}_j, \eta)$
5 : **else return** $\perp$

(a) Description of the $\mathcal{O}^{\mathsf{no\text{-}Frame}}_{\mathsf{Register}}$ and $\mathcal{O}^{\mathsf{no\text{-}Frame}}_{\mathsf{Sign}}$ Oracles.

$\mathsf{Exp}^{\mathsf{no\text{-}Frame}}_{\mathcal{A},\mathsf{k\text{-}APS}}(1^\lambda)$

1 : $\mathcal{U}, \mathcal{D}, \mathcal{S} \leftarrow \emptyset$
2 : $\mathsf{pp} \leftarrow \mathsf{Setup}_{\mathsf{k\text{-}APS}}(1^\lambda)$
3 : $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}_{\mathsf{k\text{-}APS}}(1^\lambda, k)$
4 : $(m_1^*, \sigma_1^*), (m_2^*, \sigma_2^*) \leftarrow \mathcal{A}^{\mathcal{O}^{\mathsf{no\text{-}Frame}}_{\mathsf{Register}}, \mathcal{O}_{\mathsf{Delegate}}, \mathcal{O}^{\mathsf{no\text{-}Frame}}_{\mathsf{Sign}}}(\mathsf{pk})$
5 : $(\mathsf{id}, w) \leftarrow \mathsf{Link}_{\mathsf{k\text{-}APS}}(\mathsf{pk}, m_1^*, \sigma_1^*, m_2^*, \sigma_2^*)$
6 : **if** $(\mathsf{id}, \cdot, \cdot, 1) \in \mathcal{U} \wedge |\mathcal{S}[\mathsf{id}]| \leq k$,
7 : $\quad$**return** 1
8 : **return** 0

(b) Description of the Non-frameability Experiment.

Figure 4.7: Description of Non-frameability Experiment and Oracles for k-APS.

be provided with the two linked signatures so that it can link them and produce its own token w.

In the non-frameability experiment, the adversary can request registration using the oracle $\mathcal{O}^{\mathsf{no\text{-}Frame}}_{\mathsf{Register}}$ and signatures using the oracle $\mathcal{O}^{\mathsf{no\text{-}Frame}}_{\mathsf{Sign}}$, both introduced in Figure 4.7a. A k-APS signature is *non-frameable* if for any PPT algorithm $\mathcal{A}$, the probability $\mathsf{Adv}^{\mathsf{no\text{-}Frame}}_{\mathcal{A},\mathsf{k\text{-}APS}}(1^\lambda) = \Pr[\mathsf{Exp}^{\mathsf{no\text{-}Frame}}_{\mathcal{A},\mathsf{k\text{-}APS}}(1^\lambda) = 1]$ is negligible. The security experiment $\mathsf{Exp}^{\mathsf{no\text{-}Frame}}_{\mathcal{A},\mathsf{k\text{-}APS}}(1^\lambda)$ is described in Figure 4.7b.

4.3.2 Our k-Times Anonymous Proxy Signature Scheme

In this section, we present our k-times anonymous proxy signature ($\mathsf{Setup}_{\mathsf{k\text{-}APS}}, \mathsf{Gen}_{\mathsf{k\text{-}APS}}, \mathsf{PKeyGen}_{\mathsf{k\text{-}APS}}, \mathsf{Delegate}_{\mathsf{k\text{-}APS}}, \mathsf{Sign}_{\mathsf{k\text{-}APS}}, \mathsf{Verif}_{\mathsf{k\text{-}APS}}, \mathsf{Link}_{\mathsf{k\text{-}APS}}, \mathsf{Trace}_{\mathsf{k\text{-}APS}}$), which uses a bilinear group setting $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, e)$ and a SPS scheme.

Construction Intuition. The setup (algorithm $\mathsf{Setup}_{\mathsf{k\text{-}APS}}$) of our construction returns several group elements and the description of a hash function. In particular, the group element g_1 will be used as a bases for the proxy public key $\mathsf{ppk} = g_1^{\mathsf{psk}}$ (where psk is the proxy secret key). The signer key pair (generated from based on $\mathsf{Gen}_{\mathsf{k\text{-}APS}}$) is an SPS key pair supporting vectors of $4l + 1$ group elements in $\mathbb{G}_1$, where $l = \lceil \log_2(k) \rceil$.

To delegate (algorithm $\mathsf{Delegate}_{\mathsf{k\text{-}APS}}$) the power to create k anonymous signatures, the signer will create two sets of l public/secret keys $(y_{i,0}, x_{i,0})_{i\in[\![l]\!]}$ and $(y_{i,1}, x_{i,1})_{i\in[\![l]\!]}$. The idea behind this technique is to be able to create k public/secret keys by composing the previous $2\log_2(k)$ keys: given an integer $\eta < k$, the key corresponding to η will be

composed of the keys corresponding to each of the bits in η. For each i, the signer also produce a Diffie-Hellman key $\mathsf{ppk}_{i,j} = g_1^{\mathsf{psk} \cdot x_{i,j}}$ between $y_{i,j} = g_1^{x_{i,j}}$ and $\mathsf{ppk} = g_1^{\mathsf{psk}}$. This will enable us to link the keys produced by these elements to the owner of the proxy public key ppk later on. Finally, all these public keys are signed with an SPS, acting as a certificate, so that they can be randomised, and they are stored in the delegation del. Thanks to cert, we have already shown that the proxy can produce k distinct pairs of certified Elgamal public/secret keys. The idea of our signature algorithm ($\mathsf{Sign}_{\mathsf{k\text{-}APS}}$) is to use one of its keys for each signature. If the same key is used several times, however, it will be possible to find its owner thanks to the mechanism introduced in [BL16] (this point will be discussed further in this section). However, to preserve anonymity, these keys must not be linkable, so they must be randomised (note that the SPS properties preserve their certification by the signer).

Assume that the delegate is using the algorithm for the η^{th} time. First, the delegate randomises g_1 and all the elements $y_{i,j}$ and $\mathsf{ppk}_{i,j}$ using the same random r as an exponent, and adapts the SPS accordingly. The randomised version of g_1 is denoted $\widehat{g}_1$, and the keys are respectively denoted $\widehat{y}_{i,j}$ and $\widehat{\mathsf{ppk}}_{i,j}$. This first step randomises all the elements in the delegation $y_{i,j}$ and $\mathsf{ppk}_{i,j}$, so that it is not possible to make the link between the randomised delegation $\widehat{y}_{i,j}$ and $\widehat{\mathsf{ppk}}_{i,j}$ and the original one. Then, the delegate chooses a new random s, randomises the basis $\widehat{g}_1$ in $\widetilde{g}_1$, and randomises only the $\widehat{y}_{i,\eta[i]}$ and $\widehat{\mathsf{ppk}}_{i,\eta[i]}$ corresponding to the bits of η to obtain the keys $\widetilde{y}_i$ and $\widetilde{\mathsf{ppk}}_i$. The delegate uses a zero-knowledge proof to ensure that the randomization has been done correctly and with an integer η actually lower than k (the instantiation of this proof is rather technical and described in more details in the next section). In this way, it is possible to multiply the public keys $\widetilde{y} = \prod_{i=1}^{l} \widetilde{y}_i$ and add the corresponding secret keys $x = \sum_{i=1}^{l} x_{i,\eta[i]}$ to obtain a new public/secret key pair $(\widetilde{y}, x)$ that verifies $\widetilde{y} = \widetilde{g}_1^x$. This second step allows the proxy to hide its chosen η in the elements $\widetilde{y}_i$ and $\widetilde{\mathsf{ppk}}_i$ (by randomizing the $\widehat{y}_{i,\eta[i]}$ and $\widehat{\mathsf{ppk}}_{i,\eta[i]}$ again) while preserving the link between the randomised delegation $\widehat{y}_{i,j}$ and $\widehat{\mathsf{ppk}}_{i,j}$ and the generated key pair $(\widetilde{y}, x)$. Note that $\widetilde{\mathsf{ppk}} = \prod_{i=1}^{l} \widetilde{\mathsf{ppk}}_i$ is the Diffie-Hellman of $\widetilde{y}$ and ppk, which links $\widetilde{y}$ to the owner of ppk in a hidden way. It allows the delegate to prove in zero-knowledge that the identity revealed by the mechanism of [BL16] is indeed the identity of the delegate. This proof, denoted π_σ, also proves that the mechanism of [BL16] triggers correctly if the delegate signs more than k messages. The technical description of this proof is given in Section 4.2.3.

The signature verification (algorithm $\mathsf{Verif}_{\mathsf{k\text{-}APS}}$) consists of re-computing $\widetilde{y}$ and $\widetilde{\mathsf{ppk}}$ and checking that the proof π_σ is valid. Finally, the $\mathsf{Link}_{\mathsf{k\text{-}APS}}$ and $\mathsf{Trace}_{\mathsf{k\text{-}APS}}$ algorithms work in the same way as in [BL16]: each signature contains $\alpha_1 = h_1^x$, $\alpha_2 = g_2^t$, $\alpha_3 = h_2^x \cdot g_1^{u \cdot \mathsf{psk}}$, and $\alpha_4 = h_3^x \cdot h_4^{v \cdot \mathsf{psk}}$. Thus, if the same key x is used twice in signatures, they can be linked since they share the same $\alpha_1 = h_1^x$. Let's note $\alpha'_3 = h_2^x \cdot g_1^{u' \cdot \mathsf{psk}}$ the element α_3 corresponding to the second signature. It is possible to find the identity of the delegate by computing $(\alpha_3/\alpha'_3)^{1/(u-u')} = \mathsf{ppk}$. In a similar way, the token $\omega = h_4^{\mathsf{psk}}$ leaks from α_4 when two signatures are linked. Each signature also contains elements of the form $\tau = e(\omega, \alpha_2)$. Without knowledge of ω, τ is indistinguishable from a random element under the DDH assumption, but a user which knows the delegate's token ω can retrieve its signatures by recomputing τ, thus achieving full traceability.

Formal Description. Below is the formal instantiation of our k-APS scheme.

$\underline{\mathsf{Setup}_{\mathsf{k\text{-}APS}}(1^\lambda)\colon}$ samples random base elements $g_1, h_1, h_2, h_3, h_4 \in \mathbb{G}_1$ and $g_2 \in \mathbb{G}_2$, chooses a hash function $H\colon \{0,1\}^* \to \mathbb{Z}_p^*$, and sets them as the common parameters.

$\underline{\mathsf{Gen}_{\mathsf{k\text{-}APS}}(1^\lambda, k)\colon}$ sets $l = \lceil \log_2(k) \rceil$, and returns $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}_{\mathsf{SPS}}(1^\lambda, 4l+1)$.

$\underline{\mathsf{PKeyGen}_{\mathsf{k\text{-}APS}}(1^\lambda)\colon}$ samples $\mathsf{psk} \xleftarrow{\$} \mathbb{Z}_q$ and sets $\mathsf{ppk} = g_1^{\mathsf{psk}}$. Returns the pair $(\mathsf{psk}, \mathsf{ppk})$.

$\underline{\mathsf{Delegate}_{\mathsf{k\text{-}APS}}(\mathsf{sk}, \mathsf{ppk}, k)\colon}$ sets $l = \lceil \log_2(k) \rceil$, aborts if the SPS key sk does not support message of $4l+1$ elements. For all $(i,j) \in [\![l]\!] \times \{0,1\}$, samples $x_{i,j} \xleftarrow{\$} \mathbb{Z}_p^*$, sets $y_{i,j} = g_1^{x_{i,j}}$, $\mathsf{ppk}_{i,j} = \mathsf{ppk}^{x_{i,j}}$ and produces $\widehat{\sigma} \leftarrow \mathsf{Sign}_{\mathsf{SPS}}(\mathsf{sk}, (g_1, y_{1,0}, \ldots, y_{l,1}, \mathsf{ppk}_{1,0}, \ldots, \mathsf{ppk}_{l,1}))$. Returns $\mathsf{cert} = ((x_{i,j}, y_{i,j}, \mathsf{ppk}_{i,j})_{i\in[\![l]\!];j\in\{0,1\}}, \widehat{\sigma})$.

$\underline{\mathsf{Sign}_{\mathsf{k\text{-}APS}}(\mathsf{pk}, \mathsf{psk}, m, \mathsf{cert}, \eta)\colon}$ set $l = \lceil \log_2(k) \rceil$. Samples $r, s \xleftarrow{\$} \mathbb{Z}_p^*$, sets $\widehat{g_1} = g_1^r$ and $\widetilde{g_1} = \widehat{g_1}^{\,s}$. For all $i \in [\![l]\!]$ and $j \in \{0,1\}$, computes $\widehat{y}_{i,j} = y_{i,j}^r$ and $\widehat{\mathsf{ppk}}_{i,j} = \mathsf{ppk}_{i,j}^r$, adapts the SPS signature $\widehat{\sigma}$ into an independent one $\widehat{\sigma} \leftarrow \mathsf{ChgRep}_{\mathsf{SPS}}((g_1, y_{1,0}, \ldots, y_{l,1}, \mathsf{ppk}_{1,0}, \ldots, \mathsf{ppk}_{l,1}), \widehat{\sigma}, r, \mathsf{pk})$, computes $\widetilde{y}_i = \widehat{y}_{i,\eta[i]}^s$, and $\widetilde{\mathsf{ppk}}_i = \widehat{\mathsf{ppk}}_{i,\eta[i]}^{\,s}$. Generates a zero-knowledge proof $\Pi_{<k}$ of knowledge of s and η which proves that (i) $\widetilde{y}_i$ and $\widetilde{\mathsf{ppk}}_i$ are well formed according to s and some integer η of l bits and (ii) $\eta < k$. We have presented the formalisation of this zero-knowledge proof in Section 4.2.1. Sets $x = \sum_{i=1}^{l} x_{i,\eta[i]}$, $\widetilde{y} = \prod_{i=1}^{l} \widetilde{y}_i$, $\widetilde{\mathsf{ppk}} = \prod_{i=1}^{l} \widetilde{\mathsf{ppk}}_i$, samples $t \xleftarrow{\$} \mathbb{Z}_p^*$ and computes $\alpha_1 = h_1^x$, $\alpha_2 = g_2^t$, $u = H(m, 0, \alpha_2)$ and $v = H(m, 1, \alpha_2)$. Generates the matching elements $\alpha_3 = h_2^x \cdot g_1^{u\cdot\mathsf{psk}}$ and $\alpha_4 = h_3^x \cdot h_4^{v\cdot\mathsf{psk}}$, and the tracing element $\tau = e(h_4, \alpha_2)^{\mathsf{psk}}$. Also generates:

$$\pi_\sigma \leftarrow \mathsf{ZK}\left\{ \mathsf{psk}, x, t\colon \begin{array}{l} \widetilde{y} = \widetilde{g_1}^{\,x} \wedge \widetilde{\mathsf{ppk}} = \widetilde{y}^{\mathsf{psk}} \wedge \alpha_1 = h_1^x \wedge \alpha_2 = g_2^t \\ \wedge \alpha_3 = h_2^x \cdot g_1^{u\cdot\mathsf{psk}} \wedge \alpha_4 = h_3^x \cdot h_4^{v\cdot\mathsf{psk}} \wedge \tau = e(h_4, \alpha_2)^{\mathsf{psk}} \end{array} \right\}.$$

Set $\sigma_{\mathsf{cert}} = (\widehat{g_1}, ((\widehat{y}_{i,b}, \widehat{\mathsf{ppk}}_{i,b})_{b\in\{0,1\}}, \widetilde{y}_i, \widetilde{\mathsf{ppk}}_i)_{i\in[\![l]\!]}, \widehat{\sigma}, \Pi_{<k})$ and return the signature $\sigma = (\widetilde{g_1}, \alpha_1, \alpha_2, \alpha_3, \alpha_4, \tau, \pi_\sigma, \sigma_{\mathsf{cert}})$.

$\underline{\mathsf{Verif}_{\mathsf{k\text{-}APS}}(m, \sigma, \mathsf{pk})\colon}$ parse $\sigma \xrightarrow{p} (\widetilde{g_1}, \alpha_1, \alpha_2, \alpha_3, \alpha_4, \tau, \pi_\sigma, \sigma_{\mathsf{cert}})$ and $\sigma_{\mathsf{cert}} \xrightarrow{p} (\widehat{g_1}, ((\widehat{y}_{i,b}, \widehat{\mathsf{ppk}}_{i,b})_{b\in\{0,1\}}, \widetilde{y}_i, \widetilde{\mathsf{ppk}}_i)_{i\in[\![l]\!]}, \widehat{\sigma}, \Pi_{<k})$. Compute $u = H(m, 0, \alpha_2)$, $v = H(m, 1, \alpha_2)$, $\widetilde{y}_n = \prod_{i=1}^{l} \widetilde{y}_i$, and $\widetilde{\mathsf{ppk}} = \prod_{i=1}^{l} \widetilde{\mathsf{ppk}}_i$. Verify the signature $\widehat{\sigma}$ and the proofs $\Pi_{<k}$ and π_σ. If all checks are correct, returns 1, otherwise 0

$\underline{\mathsf{Link}_{\mathsf{k\text{-}APS}}(\mathsf{pk}, m, \sigma, m', \sigma')\colon}$ verifies that $\mathsf{Verif}_{\mathsf{SPS}}(m, \sigma, \mathsf{pk}) = \mathsf{Verif}_{\mathsf{SPS}}(m', \sigma', \mathsf{pk}) = 1$ and returns 0 if $\alpha_1 \neq \alpha_1'$ or if one of the verification failed. Compute $u = H(m, 0, \alpha_2)$, $v = H(m, 1, \alpha_2)$, $u' = H(m', 0, \alpha_2')$, $v' = H(m', 1, \alpha_2')$, $\mathsf{id} = (\alpha_3/\alpha_3')^{1/(u-u')}$ and $w = (\alpha_4/\alpha_4')^{1/(v-v')}$. Returns (id, w).

$\underline{\mathsf{Trace}_{\mathsf{k\text{-}APS}}(w, \sigma)\colon}$ returns 1 if and only if $\tau = e(w, \alpha_2)$.

In the following, we informally explain why each of the security properties presented in Section 4.3 holds in our scheme.

Unforgeability. Zero-knowledge proofs produced during the signing process ensure that the delegate has used its certificate correctly. This means that a user who has not been delegated cannot produce a valid fresh signature under the assumption that the proof is sound.

Anonymity. Since the elements in the certificate are randomised for each new signature, and since the delegate is able to create public keys $\widetilde{y}$ for k different secret keys x, it is not

possible to link two signatures from the elements $\widehat{y}_{i,j}$ and $\widetilde{y}_i$ under the DDH assumption. Recall also that the $\widehat{\mathsf{ppk}}_{i,j}$ and $\widetilde{\mathsf{ppk}}_i$ do not allow the signature to be linked to ppk under the DDH assumption either. On the other hand, the element h_2^x (resp. h_3^x) is the Diffie-Hellman of h_2 (resp. h_3) and $\widetilde{y}$ (where $\widetilde{y}$ varies with each of the k signatures), and therefore hides the elements linked to the identity of the delegate including α_3 (resp. α_4). Finally, τ hides the value of ppk under the DDH assumption on the elements h_4 and ppk. Assuming that the proofs are indeed zero-knowledge, it is not possible to link two signatures from the same honest user.

Traceability and Non-frameability. Zero-knowledge proofs ensure that the signature is correct and that the delegate knows the secret key corresponding to the public key used in the certificate. Thus, the delegate cannot bypass the mechanism for linking/tracing its signatures if it exceeds the limit, which ensures traceability, and the delegate can only use elements of its own delegation, which ensures non-frameability.

We therefore have the following theorem, for which the proofs are available in [BOA24b].

Theorem 1: Security of our k-APS.

Instantiated by a signature on equivalent classes that is unforgeable, class-hiding, and signature adaptatable, by NIZK proofs which are zero-knowledge and sound, and by a collision-resistante hash function, our k-APS scheme is unforgeable, anonymous, traceable and non-frameable under the DDH assumption in $\mathbb{G}_1$ and $\mathbb{G}_2$.

4.4 k-Times Anonymous Sanitizable Signatures

Sanitizable signature schemes [ACDMT05] enable a delegate called the *sanitizer* to modify specific sections of a signed message $m = m_1 \| \dots \| m_n$ and update the signature consistently with these modifications. Sanitizable signatures can also be seen as a more restrictive variant of proxy signatures, in which the sanitizer receives delegations prescribing portions of the messages it can sign: the signer produces both a signature enforcing parts of the message and data enabling the delegate to produce new signatures using the sanitization algorithm (corresponding to the delegation certificate in the proxy signatures) as long as the restrictions chosen by the signer are respected. In this section, we bound the delegation provided to sanitizers in order to limit it to a maximum of k signatures and guarantee full traceability by the verifier in the event of the limit being exceeded.

Technical Summary

In this section we formalise the notion of a *k-times (fully traceable) anonymous sanitizable signature* (k-SAN). We first informally describe its purpose, algorithms and the security model for k-SAN. The full definitions is provided in Section 4.4.2. We then extend our proxy signature scheme to the case of sanitizable signatures in Section 4.4.2. Our formal definition combines the features of the k-times anonymous

proxy signatures defined in Section 4.3, with the standard features of sanitizable signatures [BFF+09, CJ10, KSS15, BB21]. In contrast with previous sanitizable signature models, each sanitization requires the use of a delegation that can only be used k times, even if it is used for different signatures.

4.4.1 Security Model for k-Times Anonymous Sanitizable Signatures

Highlight of the Model. A k-SAN consists of the following algorithms: $\mathsf{Setup}_{\text{k-SAN}}$, $\mathsf{Gen}_{\text{k-SAN}}$, $\mathsf{SaKeyGen}_{\text{k-SAN}}$, $\mathsf{Delegate}_{\text{k-SAN}}$, $\mathsf{Sign}_{\text{k-SAN}}$, $\mathsf{Sanitize}_{\text{k-SAN}}$, $\mathsf{Verif}_{\text{k-SAN}}$, $\mathsf{Link}_{\text{k-SAN}}$ and $\mathsf{Trace}_{\text{k-SAN}}$. With the exception of $\mathsf{Sign}_{\text{k-SAN}}$ and $\mathsf{Sanitize}_{\text{k-SAN}}$, all these algorithms are defined in a similar way to k-APS ($\mathsf{SaKeyGen}_{\text{k-SAN}}$ corresponds to $\mathsf{PKeyGen}_{\text{k-APS}}$ and generates the sanitizer key pair $(\mathsf{ssk}, \mathsf{spk})$). The only difference with the previously described $\mathsf{Sign}_{\text{k-SAN}}$ and $\mathsf{Sanitize}_{\text{k-SAN}}$ algorithms is that the modification of the authorised message must be prescribed.

A k-times Anonymous Sanitizable Signature scheme is required to achieve *Unforgeability*, *Immutability*, *Transparency*, *Invisibility*, *Unlinkability*, *Anonymity*, *Traceability* and *Non-frameability*.

Note that sanitizable signatures usually have two additional algorithms, Prove and Judge, which allow the delegating signer to reveal *a posteriori* that a given signature was produced by the sanitizer. In this case, an additional security property, accountability, is required to ensure that the signer cannot blame the sanitizer for a signature it did not produce, and that the sanitizer will not be able to produce a signature that cannot be traced by the signer. Since in this chapter we are considering a scenario where the tracing of dishonest users is not done by the signer, but by the verifier using the mechanism triggered when the sanitizer produces too many signatures, we have not provided our construction with these algorithms and have not adapted the accountability model.

Formal Definition. We now fully detail the model for k-times anonymous sanitizable signature schemes.

Definition 28: k-times Anonymous Sanitizable Signature scheme - k-SAN

A *k-times Anonymous Sanitizable Signature scheme* (k-SAN) is a tuple of polynomial-time algorithms:

$\mathsf{Setup}_{\text{k-SAN}}(1^\lambda)$: given a security parameter, returns public parameters pp. Note that pp is assumed to be an implicit input to all the following algorithms.

$\mathsf{Gen}_{\text{k-SAN}}(1^\lambda, k, n)$: given a security parameter and two integers k and n, returns a pair of key $(\mathsf{sk}, \mathsf{pk})$.

$\mathsf{SaKeyGen}_{\text{k-SAN}}(1^\lambda)$: given a security parameter 1^λ, returns a pair of keys $(\mathsf{ssk}, \mathsf{spk})$.

$\mathsf{Delegate}_{\text{k-SAN}}(\mathsf{sk}, \mathsf{spk}, k)$: given the keys sk and spk and an integer k, returns a delegation del.

$\mathsf{Sign}_{\mathsf{k\text{-}SAN}}(m, \mathsf{ADM}, \mathsf{sk}, \mathsf{spk})$: given the keys sk and spk, a message $m = m_1 \| \dots \| m_n$, for a given index n, and a admissible set $\mathsf{ADM} \subset [\![n]\!]$, returns a signature σ.

$\mathsf{Sanitize}_{\mathsf{k\text{-}SAN}}(m, \sigma, \mathsf{MOD}, \mathsf{ssk}, \mathsf{pk}, \mathsf{del}, \eta)$: given the keys pk and ssk, a message-signature pair (m, σ), a modification MOD, a delegation del and a signature index η, returns a signature σ'.

$\mathsf{Verif}_{\mathsf{k\text{-}SAN}}(\mathsf{pk}, m, \sigma)$: given a key pk, a message m and a signature σ, the algorithm returns 0 or 1.

$\mathsf{Link}_{\mathsf{k\text{-}SAN}}(\mathsf{pk}, m, \sigma, m', \sigma')$: given a key pk, two message-signature pair m, σ and m', σ', returns an identity id and a witness w or the special symbol $\perp$.

$\mathsf{Trace}_{\mathsf{k\text{-}SAN}}(w, \sigma)$: given a witness w and a signature σ, returns 0 or 1.

We require that k-SAN meets *Unforgeability*, *Immutability*, *Transparency*, *Invisibility*, *Unlinkability*, *Anonymity*, *Traceability* and *Non-frameability* as defined hereafter.

For the following definitions, we provide two notations ADM and MOD, which are respectively the admissible modification for the message associated to the signature produced by the signer and the modification desired by the sanitizer. ADM is defined as a subset of $[\![n]\!]$ for a message consisting of n parts in Definition 28. In order to simplify notations, we will also use ADM as a function taking as input a given modification MOD and returning 1 if these modifications are admissible, *i.e.,* if $\mathsf{MOD}(m)$ has modified only blocks of the message whose index is in the set ADM. Otherwise, the value for $\mathsf{ADM}(\mathsf{MOD})$ is 0.

The security properties of sanitizable signatures have already been investigated in numerous previous works [ACDMT05, BFF+09, CJ10, BB21]. We have adapted the existing security properties to the newly introduced model. We also add the properties related to the k-times mechanism: *anonymity*, *traceability* and *non-frameability*. These properties stay consistent with what was defined for proxy signatures (Section 4.3), as both type of signatures share conceptual similarities, but diverge in practical usages.

Before stating the security experiments, we begin the presentation of the properties of k-times Anonymous Sanitizable Signature schemes with the description of the common oracles used to formalise the properties of immutability, transparency, invisibility, unlinkability and anonymity.

$\mathcal{O}_{\mathsf{del}}(\mathsf{sk}, \underline{\mathsf{spk}, l \leq k})$	$\mathcal{O}_{\mathsf{Sign}}(\mathsf{sk}, \underline{\mathsf{spk}, m, \mathsf{ADM}})$
1 : $\mathsf{del} \leftarrow \mathsf{Delegate}_{\mathsf{k\text{-}SAN}}(\mathsf{sk}, \mathsf{spk}, l)$	1 : $\sigma \leftarrow \mathsf{Sign}_{\mathsf{k\text{-}SAN}}(m, \mathsf{ADM}, \mathsf{sk}, \mathsf{spk})$
2 : **return** del	2 : $\mathcal{S} \leftarrow \mathcal{S} \cup \{(m, \sigma, \mathsf{ADM}, \mathsf{spk})\}$
	3 : **return** σ

Figure 4.8: Description of the $\mathcal{O}_{\mathsf{del}}$ and $\mathcal{O}_{\mathsf{Sign}}$ Oracles for k-SAN.

Unforgeability. Users cannot generate a valid signature without knowing a secret key which has obtained a delegation. A k-times anonymous sanitizable signature is *unforgeable* when for any PPT adversary $\mathcal{A}$, the probability that $\mathcal{A}$ wins the SUF experiment

is negligible for every $n \in \mathbb{N}$. The security experiment $\mathsf{Exp}^{\mathsf{SUF}}_{\mathcal{A},\mathsf{k\text{-}SAN}}(1^\lambda)$ is described in Figure 4.9b. In the unforgeability experiment, the adversary can request signatures from

$\mathcal{O}^{\mathsf{SUF/unlink}}_{\mathsf{Sign}}(\mathsf{sk},\mathsf{spk},\underline{m},\mathsf{ADM})$	$\mathcal{O}^{\mathsf{SUF}}_{\mathsf{San}}(\mathsf{ssk},\mathsf{pk},\mathsf{del},\underline{m,\sigma,\mathsf{MOD}},\eta)$
1 : $\sigma \leftarrow \mathsf{Sign}_{\mathsf{k\text{-}SAN}}(m,\mathsf{ADM},\mathsf{sk},\mathsf{spk})$	1 : $\sigma \leftarrow \mathsf{Sanitize}_{\mathsf{k\text{-}SAN}}(m,\sigma,\mathsf{MOD},\mathsf{ssk},\mathsf{pk},\mathsf{del},\eta)$
2 : $\mathcal{S} \leftarrow \mathcal{S} \cup \{(m,\sigma,\mathsf{ADM},\mathsf{spk})\}$	2 : $\mathcal{S} \leftarrow \mathcal{S} \cup \{(\mathsf{MOD}(m),\sigma)\}$
3 : **return** σ	3 : **return** σ

(a) Description of the $\mathcal{O}^{\mathsf{SUF/unlink}}_{\mathsf{Sign}}$ and $\mathcal{O}^{\mathsf{SUF}}_{\mathsf{San}}$ Oracles.

$\mathsf{Exp}^{\mathsf{SUF}}_{\mathcal{A},\mathsf{k\text{-}SAN}}(1^\lambda, n)$

1 : $\mathcal{S} \leftarrow \emptyset$
2 : $\mathsf{pp} \leftarrow \mathsf{Setup}_{\mathsf{k\text{-}SAN}}(1^\lambda)$
3 : $(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{Gen}_{\mathsf{k\text{-}SAN}}(1^\lambda, k, n)$
4 : $(\mathsf{spk},\mathsf{ssk}) \leftarrow \mathsf{SaKeyGen}_{\mathsf{k\text{-}SAN}}(1^\lambda)$
5 : $\mathsf{del} \leftarrow \mathsf{Delegate}_{\mathsf{k\text{-}SAN}}(\mathsf{sk},\mathsf{spk},k)$
6 : $(m^*,\sigma^*) \leftarrow \mathcal{A}^{\mathcal{O}^{\mathsf{SUF/unlink}}_{\mathsf{Sign}},\mathcal{O}^{\mathsf{SUF}}_{\mathsf{San}}}(\mathsf{pk},\mathsf{spk})$
7 : **if** $\exists \mathsf{ADM},\mathsf{spk},(m^*,\sigma^*,\mathsf{ADM},\mathsf{spk}) \notin \mathcal{S}$: **return** $\mathsf{Verif}_{\mathsf{k\text{-}SAN}}(m^*,\sigma^*,\mathsf{pk})$
8 : **return** 0

(b) Description of the Unforgeability Experiment.

Figure 4.9: Description of Unforgeability Experiment and Oracles for k-SAN.

the signer based on $\mathcal{O}^{\mathsf{SUF/unlink}}_{\mathsf{Sign}}$ and sanitizations from the targeted sanitizer via $\mathcal{O}^{\mathsf{SUF}}_{\mathsf{San}}$, both introduced in Figure 4.9a. No signature limit is imposed, as revealing the identity of the signer after too many signatures should not affect the unforgeability of the sanitizable signature scheme. Also, due to the anonymity of the signature, this experiment is modelled without providing delegations to the adversary, otherwise a signature verifying $\mathsf{Verif}_{\mathsf{k\text{-}SAN}}(m^*,\sigma^*,\mathsf{pk})$ would be easily produced by the adversary based on other signature keys which have received a delegation. This is because signatures from different sanitizers are expected to be *indistinguishable*, according to the property referred under this name.

Immutability. A sanitizable signature is *immutable* when no adversary is able to sanitize a message by means of making unauthorised modifications. A k-times anonymous sanitizable signature is *immutable* when for any PPT time adversary $\mathcal{A}$, the probability that $\mathcal{A}$ wins the Immut experiment is negligible for every $n \in \mathbb{N}$. The security experiment $\mathsf{Exp}^{\mathsf{Immut}}_{\mathcal{A},\mathsf{k\text{-}SAN}}(1^\lambda)$ is described in Figure 4.10 on the following page. The adversary has access to a delegation oracle $\mathcal{O}_{\mathsf{del}}$ and a signature oracle $\mathcal{O}_{\mathsf{Sign}}$ described in Figure 4.8 on the preceding page.

Common Experiment for Multiple Properties. We strucutre the *transparency*, *invisibility* and *unlinkability* in the same generic way, but using different oracles. The security experiment $\mathsf{Exp}^{\mathcal{O}\text{-}\mathsf{Sanitize}}_{\mathcal{A},\mathsf{k\text{-}SAN}}(1^\lambda)$ is described in Figure 4.11 on the following page and mainly follows the direct execution of algorithms. It starts by sampling a random value $b \xleftarrow{\$} \{0,1\}$ and setting up the environment via the Setup algorithm. It then generates

$\mathsf{Exp}^{\mathsf{Immut}}_{\mathcal{A},\mathsf{k\text{-}SAN}}(1^\lambda, n)$

$\mathcal{S} \leftarrow \emptyset$
$\mathsf{pp} \leftarrow \mathsf{Setup}_{\mathsf{k\text{-}SAN}}(1^\lambda)$
$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}_{\mathsf{k\text{-}SAN}}(1^\lambda, k, n)$
$(m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{del}}, \mathcal{O}_{\mathsf{Sign}}}(\mathsf{pk})$
if $(\mathsf{Verif}_{\mathsf{k\text{-}SAN}}(m^*, \sigma^*, \mathsf{pk}) = 1) \wedge$
$(\forall\ \mathsf{MOD}, \forall (m, \sigma, \mathsf{ADM}, \mathsf{spk}) \in \mathcal{S}$ s.t. $\mathsf{ADM}(\mathsf{MOD}) = 1, m^* \neq \mathsf{MOD}(m)) :$
  return 1
return 0

Figure 4.10: Description of the Traceability Experiment for k-SAN.

the signer's key pair $(\mathsf{pk}, \mathsf{sk})$, the sanitizer's key pair $(\mathsf{spk}, \mathsf{ssk})$ and a delegation del. Once these elements are produced, the adversary obtains the public keys pk and spk and accesses the oracles in an oracle list $\mathcal{O}$. Amongst these oracles, the challenge oracles are those related to the winning condition. The adversary is asked to produce a decision bit b^* based on the challenges accessed via one of the oracles. The experiment produces a value that depends on the equality of the bits b and b^*.

$\mathsf{Exp}^{\mathcal{O}\text{-}\mathsf{Sanitize}}_{\mathcal{A},\mathsf{k\text{-}SAN}}(1^\lambda, n)$

$\mathcal{S}, \mathcal{H} \leftarrow \emptyset, b \xleftarrow{\$} \{0,1\}$ // defined as global variables
$\mathsf{pp} \leftarrow \mathsf{Setup}_{\mathsf{k\text{-}SAN}}(1^\lambda)$
$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}_{\mathsf{k\text{-}SAN}}(1^\lambda, k, n)$
$(\mathsf{spk}, \mathsf{ssk}) \leftarrow \mathsf{SaKeyGen}_{\mathsf{k\text{-}SAN}}(1^\lambda)$
$\mathsf{del} \leftarrow \mathsf{Delegate}_{\mathsf{k\text{-}SAN}}(\mathsf{sk}, \mathsf{spk}, k)$
$b^* \leftarrow \mathcal{A}^{\mathcal{O}}(\mathsf{pk}, \mathsf{spk})$
return $b = b^*$

Figure 4.11: Description of the $\mathcal{O}$-Sanitize Experiment for k-SAN.
Generic Experiment for Transparency, Invisibility and Unlinkability.

Transparency. The verifier cannot decide whether a given signature has been sanitized or not. A k-times anonymous sanitizable signature is *transparent* when for any PPT adversary $\mathcal{A}$, the probability that $\mathcal{A}$ wins the $\{\mathcal{O}^{\mathsf{tran}}_{\mathsf{Sa/Si}}, \mathcal{O}_{\mathsf{del}}, \mathcal{O}_{\mathsf{Sign}}, \mathcal{O}^{\mathsf{tran}}_{\mathsf{San}}\}$-Sanitize experiment is negligible for every $n \in \mathbb{N}$. $\mathsf{Exp}^{\mathcal{O}\text{-}\mathsf{Sanitize}}_{\mathcal{A},\mathsf{k\text{-}SAN}}(1^\lambda)$ is described in Figure 4.11. The adversary has access to a delegation oracle $\mathcal{O}_{\mathsf{del}}$ and a signature oracle $\mathcal{O}_{\mathsf{Sign}}$ described in Figure 4.8 on page 100. It also has access to the oracle $\mathcal{O}^{\mathsf{tran}}_{\mathsf{San}}$ which provides the sanitization of the experiment by the sanitizer and to the challenge oracle $\mathcal{O}^{\mathsf{tran}}_{\mathsf{Sa/Si}}$ which produces either signatures from the signer (case $b = 0$) or signatures sanitized by the sanitizer (case $b = 1$). These last two oracles are described in Figure 4.12 on the next page.

Invisibility. The invisibility property prevents an adversary which is neither the signer, nor the sanitizer of a signature from determining any information on the modifiable

$\mathcal{O}^{\mathsf{tran}}_{\mathsf{Sa/Si}}(\mathsf{sk}, \mathsf{ssk}, \mathsf{del}, \underline{m, \mathsf{ADM}, \mathsf{MOD}, \eta})$

1 : **if** $\mathsf{ADM}(\mathsf{MOD}) = 0 \vee \eta \in \mathcal{H} \vee \eta \geq k$:
2 : **return** $\perp$
3 : **if** $b = 0$:
4 : $\sigma \leftarrow \mathsf{Sign}_{\mathsf{k\text{-}SAN}}(m, \mathsf{ADM}, \mathsf{sk}, \mathsf{spk})$
5 : $\sigma \leftarrow \mathsf{Sanitize}_{\mathsf{k\text{-}SAN}}(m, \sigma, \mathsf{MOD}, \mathsf{ssk}, \mathsf{pk}, \mathsf{del}, \eta)$
6 : **if** $b = 1$:
7 : $\sigma \leftarrow \mathsf{Sign}_{\mathsf{k\text{-}SAN}}(\mathsf{MOD}(m), \mathsf{ADM}, \mathsf{sk}, \mathsf{spk})$
8 : $\mathcal{H} \leftarrow \mathcal{H} \cup \{\eta\}$
9 : **return** σ

$\mathcal{O}^{\mathsf{tran}}_{\mathsf{San}}(\mathsf{ssk}, \mathsf{pk}, \mathsf{del}, \underline{m, \sigma, \mathsf{MOD}, \eta})$

1 : **if** $\mathsf{ADM}(\mathsf{MOD}) = 0 \vee \eta \in \mathcal{H} \vee \eta \geq k$:
2 : **return** $\perp$
3 : $\sigma \leftarrow \mathsf{Sanitize}_{\mathsf{k\text{-}SAN}}(m, \sigma, \mathsf{MOD}, \mathsf{ssk}, \mathsf{pk}, \mathsf{del}, \eta)$
4 : $\mathcal{H} \leftarrow \mathcal{H} \cup \{\eta\}$
5 : **return** σ

Figure 4.12: Description of the $\mathcal{O}^{\mathsf{tran}}_{\mathsf{Sa/Si}}$ and $\mathcal{O}^{\mathsf{tran}}_{\mathsf{San}}$ Oracles for Transparency of k-SAN.

blocks. More formally, a k-times anonymous sanitizable signature is *invisible* when for any PPT adversary $\mathcal{A}$, the probability that $\mathcal{A}$ wins the $\{\mathcal{O}^{\mathsf{Invis}}_{\mathsf{LRADM}}, \mathcal{O}_{\mathsf{del}}, \mathcal{O}_{\mathsf{Sign}}, \mathcal{O}^{\mathsf{Invis}}_{\mathsf{San}}\}$-Sanitize experiment is negligible for every $n \in \mathbb{N}$. The security experiment $\mathsf{Exp}^{\mathcal{O}\text{-Sanitize}}_{\mathcal{A},\mathsf{k\text{-}SAN}}(1^\lambda)$ is described in Figure 4.11 on the facing page. The adversary has access to a delegation oracle $\mathcal{O}_{\mathsf{del}}$, and a signature oracle $\mathcal{O}_{\mathsf{Sign}}$ described in Figure 4.8. It also has access to the oracle $\mathcal{O}^{\mathsf{Invis}}_{\mathsf{San}}$, which provides the sanitization of the experiment by the sanitizer, and to the challenge oracle $\mathcal{O}^{\mathsf{Invis}}_{\mathsf{LRADM}}$ which is a *Left-or-Right* oracle producing either signatures for an admissible set ADM_0 of for a an admissible set ADM_1, depending on the value of the bit b which has to be guessed by the adverary. These last two oracles are described in Figure 4.13.

$\mathcal{O}^{\mathsf{Invis}}_{\mathsf{LRADM}}(\mathsf{sk}, \mathsf{spk}, \underline{m, \mathsf{ADM}_0, \mathsf{ADM}_1})$

1 : $\sigma \leftarrow \mathsf{Sign}_{\mathsf{k\text{-}SAN}}(m, \mathsf{ADM}_b, \mathsf{sk}, \mathsf{spk})$
2 : $\mathcal{S} \leftarrow \mathcal{S} \cup \{(m, \sigma, \mathsf{ADM}_0 \cap \mathsf{ADM}_1, \mathsf{spk})\}$
3 : **return** σ

$\mathcal{O}^{\mathsf{Invis}}_{\mathsf{San}}(\mathsf{ssk}, \mathsf{pk}, \mathsf{del}, \underline{m, \sigma, \mathsf{MOD}, \eta})$

1 : **if** for some ADM, $((m, \sigma, \mathsf{ADM}, \mathsf{spk}) \in \mathcal{S})$
2 : $\wedge\, (\mathsf{ADM}(\mathsf{MOD}) = 0)$: **return** $\perp$
3 : **if** $\mathsf{Verif}_{\mathsf{k\text{-}SAN}}(m, \sigma, \mathsf{pk}) = 0$, **return** $\perp$
4 : $\sigma \leftarrow \mathsf{Sanitize}_{\mathsf{k\text{-}SAN}}(m, \sigma, \mathsf{MOD}, \mathsf{ssk}, \mathsf{pk}, \mathsf{del}, \eta)$
5 : $\mathcal{S} \leftarrow \mathcal{S} \cup \{(\mathsf{MOD}(m), \sigma, \mathsf{ADM}, \mathsf{spk})\}$
6 : **return** σ

Figure 4.13: Description of the $\mathcal{O}^{\mathsf{Invis}}_{\mathsf{LRADM}}$ and $\mathcal{O}^{\mathsf{Invis}}_{\mathsf{San}}$ Oracles for Invisibility of k-SAN.

Unlinkability. Considering a fixed sanitizer assigned with two signatures, the verifier cannot link a sanitized signature with its original version. A k-times anonymous sanitizable signature is *unlinkable* when for any PPT adversary $\mathcal{A}$, the probability that $\mathcal{A}$ wins the $\{\mathcal{O}^{\mathsf{unlink}}_{\mathsf{LRSan}}, \mathcal{O}_{\mathsf{del}}, \mathcal{O}^{\mathsf{SUF/unlink}}_{\mathsf{Sign}}, \mathcal{O}^{\mathsf{unlink}}_{\mathsf{San}}\}$-Sanitize experiment is negligible for every $n \in \mathbb{N}$. $\mathsf{Exp}^{\mathcal{O}\text{-Sanitize}}_{\mathcal{A},\text{k-SAN}}(1^\lambda)$ is described in Figure 4.11 on page 102. The adversary has access to a delegation oracle $\mathcal{O}_{\mathsf{del}}$ described in Figure 4.8. It also has access to the oracle $\mathcal{O}^{\mathsf{SUF/unlink}}_{\mathsf{Sign}}$, which provides signatures for any provided messages and admissible set ADM and to the $\mathcal{O}^{\mathsf{unlink}}_{\mathsf{San}}$, which provides the sanitization by the sanitizer of the experiment. The challenge oracle $\mathcal{O}^{\mathsf{unlink}}_{\mathsf{LRSan}}$ is a *Left-or-Right* oracle, producing signatures for either an admissible set ADM_0 of for a an admissible set ADM_1, depending on the value of the bit b which has to be guessed by the adverary. These last two oracles are described in Figure 4.14.

$\mathcal{O}^{\mathsf{unlink}}_{\mathsf{LRSan}}(\mathsf{ssk}, \mathsf{pk}, \mathsf{del}, \underline{(m_i, \mathsf{MOD}_i, \sigma_i)_{i\in\{0,1\}}}, \eta)$

1 : **if** $\exists i \in \{0,1\}, \mathsf{ADM}_i(\mathsf{MOD}_i) = 0 \vee$
2 : $\quad \exists i \in \{0,1\}, \mathsf{Verif}_{\text{k-SAN}}(m_i, \sigma_i, \mathsf{pk}) = 0$
3 : $\quad \vee\, \mathsf{ADM}_0 \neq \mathsf{ADM}_1 \vee \mathsf{MOD}_0(m_0) \neq \mathsf{MOD}_1(m_1)$
4 : $\quad \vee\, \eta \in \mathcal{H} \vee \eta \geq k$: **return** $\perp$
5 : $\sigma \leftarrow \mathsf{Sanitize}_{\text{k-SAN}}(m_b, \sigma_b, \mathsf{MOD}_b, \mathsf{ssk}, \mathsf{pk}, \mathsf{del}, \eta)$
6 : $\mathcal{S} \leftarrow \mathcal{S} \cup \{(\mathsf{MOD}_b(m_b), \sigma, \mathsf{ADM}_b, \mathsf{spk})\}$
7 : $\mathcal{H} \leftarrow \mathcal{H} \cup \{\eta\}$
8 : **return** σ

$\mathcal{O}^{\mathsf{unlink}}_{\mathsf{San}}(\mathsf{ssk}, \mathsf{pk}, \mathsf{del}, \underline{m, \sigma, \mathsf{MOD}, \eta})$

1 : **if** $\mathsf{ADM}(\mathsf{MOD}) = 0 \vee \eta \in \mathcal{H} \vee \eta \geq k$: **return** $\perp$
2 : **if** $((m, \sigma, \mathsf{ADM}, \mathsf{spk}) \in \mathcal{S}) \wedge (\mathsf{ADM}(\mathsf{MOD}) = 0,$
3 : $\quad$ for some $\mathsf{ADM})$: **return** $\perp$
4 : $\sigma \leftarrow \mathsf{Sanitize}_{\text{k-SAN}}(m, \sigma, \mathsf{MOD}, \mathsf{ssk}, \mathsf{pk}, \mathsf{del}, \eta)$
5 : $\mathcal{S} \leftarrow \mathcal{S} \cup \{(\mathsf{MOD}(m), \sigma, \mathsf{ADM}, \mathsf{spk})\}$
6 : $\mathcal{H} \leftarrow \mathcal{H} \cup \{\eta\}$
7 : **return** σ

Figure 4.14: Description of the $\mathcal{O}^{\mathsf{unlink}}_{\mathsf{San}}$ and $\mathcal{O}^{\mathsf{unlink}}_{\mathsf{LRSan}}$ Oracles for Unlinkability of k-SAN.

Anonymity. Upon receiving a delegation for k sanitizations, a sanitizer leaks its identity only if it sanitizes more than k signatures (this is guaranteed by the upcoming property of traceability), otherwise, it is not possible to link a signature to a sanitizer. A k-times anonymous sanitizable signature is *anonymous* when for any PPT adversary $\mathcal{A}$, the probability that $\mathcal{A}$ wins the ano experiment is negligible for every $n \in \mathbb{N}$. The security experiment $\mathsf{Exp}^{ano}_{\mathcal{A},\text{k-SAN}}(1^\lambda)$ is described in Figure 4.15b on the facing page.

The experiment allows the adversary to choose the index $t \in [\![k]\!]$ such that, for any number of delegated signatures, the adversary should remain unable to distinguish the challenges provided on the basis of two challenge oracles: *a)* oracle $\mathcal{O}^{\mathsf{ano}}_{\mathsf{chal\text{-}Sign}}$ which provides a signature that can only be sanitized by the challenger sanitizer, *b)* oracle $\mathcal{O}^{\mathsf{ano}}_{\mathsf{chal\text{-}San}}$ which can only be requested based on the signature previously produced by the oracle $\mathcal{O}^{\mathsf{ano}}_{\mathsf{chal\text{-}Sign}}$ and generates a signature sanitized by the challenger's sanitizer.

$\mathcal{O}^{\mathsf{ano}}_{\mathsf{San}}(\mathsf{pk}, (\mathsf{ssk}_i, \mathsf{del}_i)_{i\in\{0,1\}}, \underline{j, m, \sigma, \mathsf{MOD}})$

1 : **if** $\nexists \mathsf{ADM}, (m, \sigma, \mathsf{ADM}, \mathsf{spk}_j) \in \mathcal{S}$, s.t. $(\mathsf{ADM}(\mathsf{MOD}) = 1)$:
2 : **return** $\perp$
3 : $\sigma \leftarrow \mathsf{Sanitize}_{\mathsf{k\text{-}SAN}}(m, \sigma, \mathsf{MOD}, \mathsf{ssk}_j, \mathsf{pk}, \mathsf{del}_j, \eta_j)$
4 : $\mathcal{S} \leftarrow \mathcal{S} \cup \{(\mathsf{MOD}(m), \sigma, \mathsf{ADM}, \mathsf{spk}_j)\}$
5 : $\eta_j \leftarrow \eta_j + 1$
6 : **return** σ

$\mathcal{O}^{\mathsf{ano}}_{\mathsf{chal\text{-}Sign}}(\mathsf{sk}, (\mathsf{ssk}_i, \mathsf{del}_i)_{i\in\{0,1\}}, \underline{m, \mathsf{ADM}})$

1 : $\sigma \leftarrow \mathsf{Sign}_{\mathsf{k\text{-}SAN}}(m, \mathsf{ADM}, \mathsf{sk}, \mathsf{spk}_b)$
2 : $\mathcal{S}_{\mathsf{chal}} \leftarrow \mathcal{S}_{\mathsf{chal}} \cup \{(\mathsf{MOD}(m), \sigma, \mathsf{ADM})\}$
3 : **return** σ

$\mathcal{O}^{\mathsf{ano}}_{\mathsf{chal\text{-}San}}(\mathsf{sk}, (\mathsf{ssk}_i, \mathsf{del}_i)_{i\in\{0,1\}}, \underline{m, \sigma, \mathsf{MOD}})$

1 : **if** $\nexists \mathsf{ADM}, (m, \sigma, \mathsf{ADM}) \in \mathcal{S}_{\mathsf{chal}}$, s.t. $(\mathsf{ADM}(\mathsf{MOD}) = 1)$:
2 : **return** $\perp$
3 : $\sigma \leftarrow \mathsf{Sanitize}_{\mathsf{k\text{-}SAN}}(m, \sigma, \mathsf{MOD}, \mathsf{ssk}_b, \mathsf{pk}, \mathsf{del}_b, \eta_b)$
4 : $\mathcal{S}_{\mathsf{chal}} \leftarrow \mathcal{S}_{\mathsf{chal}} \cup \{(\mathsf{MOD}(m), \sigma, \mathsf{ADM})\}$
5 : $\eta_b \leftarrow \eta_b + 1, \gamma \leftarrow \gamma + 1$
6 : **return** σ

(a) Description of the $\mathcal{O}^{\mathsf{ano}}_{\mathsf{San}}$, $\mathcal{O}^{\mathsf{ano}}_{\mathsf{chal\text{-}Sign}}$ and $\mathcal{O}^{\mathsf{ano}}_{\mathsf{chal\text{-}San}}$ Oracles for Anonymity of k-SAN.

$\mathsf{Exp}^{\mathsf{ano}}_{\mathcal{A},\mathsf{k\text{-}SAN}}(1^\lambda, n)$

1 : $b \xleftarrow{\$} \{0,1\}, \eta_0, \eta_1 \leftarrow 0, \gamma \leftarrow 0, \mathcal{S}, \mathcal{S}_{\mathsf{chal}} \leftarrow \emptyset$ // defined as global variables
2 : $\mathsf{pp} \leftarrow \mathsf{Setup}_{\mathsf{k\text{-}SAN}}(1^\lambda)$
3 : $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}_{\mathsf{k\text{-}SAN}}(1^\lambda, k, n)$
4 : **for** $j \in \{0,1\}$,
5 : $(\mathsf{spk}_j, \mathsf{ssk}_j) \leftarrow \mathsf{SaKeyGen}_{\mathsf{k\text{-}SAN}}(1^\lambda)$
6 : $t \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{del}}, \mathcal{O}_{\mathsf{Sign}}}(\mathsf{pk}, \mathsf{spk}_0, \mathsf{spk}_1)$
7 : **if** $t \notin [\![k]\!]$, **return** b
8 : **for** $j \in \{0,1\}$,
9 : $\mathsf{del}_j \leftarrow \mathsf{Delegate}_{\mathsf{k\text{-}SAN}}(\mathsf{sk}, \mathsf{spk}_j, t)$
10 : $b^* \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{del}}, \mathcal{O}_{\mathsf{Sign}}, \mathcal{O}^{\mathsf{ano}}_{\mathsf{San}}, \mathcal{O}^{\mathsf{ano}}_{\mathsf{chal\text{-}Sign}}, \mathcal{O}^{\mathsf{ano}}_{\mathsf{chal\text{-}San}}}(\mathsf{pk}, \mathsf{spk}_0, \mathsf{spk}_1)$
11 : **if** $\eta_b \geq t \vee \eta_{b-1} \geq t\text{-}\gamma$, **return** b
12 : **return** $b = b^*$

(b) Description of the Anonymity Experiment.

Figure 4.15: Description of Anonymity Experiment and Oracles for k-SAN.

These two oracles are described in Figure 4.15a on the previous page. The adversary also has access to the delegation oracle $\mathcal{O}_{\mathsf{del}}$ and the signature oracle $\mathcal{O}_{\mathsf{Sign}}$ described in Figure 4.8 on page 100, as well as to the sanitization oracle $\mathcal{O}_{\mathsf{San}}^{\mathsf{ano}}$, which sanitizes the signature produced by the oracle $\mathcal{O}_{\mathsf{Sign}}$. These last two oracles are described in Figure 4.15a on the previous page.

Traceability. Whenever an adversary exceeds the sanitization bound, it must be possible to trace all the signatures generated by it. A k-times anonymous sanitizable signature is *traceable* when, for any PPT adversary $\mathcal{A}$, the probability that $\mathcal{A}$ wins the Trace experiment based on the CheckTrace algorithm of Figure 4.5 on page 93 is negligible for every $n \in \mathbb{N}$. The security experiment $\mathsf{Exp}^{\mathsf{Trace}}_{\mathcal{A},\mathsf{k\text{-}SAN}}(1^\lambda)$ is described in Figure 4.16b. The adversary also has access to a signing oracle $\mathcal{O}_{\mathsf{Sign}}$ described in Figure 4.8, and a delegation oracle described in Figure 4.16a.

$\mathcal{O}^{\mathsf{trace}}_{\mathsf{del}}(\mathsf{sk}, \underline{\mathsf{spk}, l \leq k})$

1 : $\mathsf{del} \leftarrow \mathsf{Delegate}_{\mathsf{k\text{-}SAN}}(\mathsf{sk}, \mathsf{spk}, l)$
2 : $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\mathsf{spk}, \mathsf{del}, l)\}$
3 : **return** del

(a) Description of the $\mathcal{O}^{\mathsf{trace}}_{\mathsf{del}}$ Oracle for Traceability of k-SAN.

$\mathsf{Exp}^{\mathsf{Trace}}_{\mathcal{A},\mathsf{k\text{-}SAN}}(1^\lambda, n)$

1 : $\mathcal{S}, \mathcal{D} \leftarrow \emptyset$
2 : $\mathsf{pp} \leftarrow \mathsf{Setup}_{\mathsf{k\text{-}SAN}}(1^\lambda)$
3 : $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}_{\mathsf{k\text{-}SAN}}(1^\lambda, k, n)$
4 : $(m_i^*, \sigma_i^*)_{i=1}^{q_s} \leftarrow \mathcal{A}^{\mathcal{O}^{\mathsf{trace}}_{\mathsf{del}}, \mathcal{O}_{\mathsf{Sign}}}(\mathsf{pk})$
5 : **return** $\mathsf{CheckTrace}(\mathsf{pk}, (m_i^*, \sigma_i^*)_{i=1}^{q_s})$

(b) Description of the Traceability Experiment.

Figure 4.16: Description of Traceability Experiment and Oracle for k-SAN.

Non-frameability. This property prevents an adversary from framing someone else by generating malformed, yet valid sanitizations. A k-times anonymous sanitizable signature is *non-frameable* when, for any PPT adversary $\mathcal{A}$, the probability that $\mathcal{A}$ wins the no-Frame experiment is negligible for every $n \in \mathbb{N}$. The security experiment $\mathsf{Exp}^{\mathsf{no\text{-}Frame}}_{\mathcal{A},\mathsf{k\text{-}SAN}}(1^\lambda)$ is described in Figure 4.17b on the facing page. The adversary has access to four oracles: a registration oracle $\mathcal{O}^{\mathsf{no\text{-}Frame}}_{\mathsf{Register}}$, generating keys for new sanitizers, a delegation oracle $\mathcal{O}^{\mathsf{no\text{-}Frame}}_{\mathsf{del}}$ providing delegation from the signer to any sanitizer, a signature oracle $\mathcal{O}_{\mathsf{Sign}}$, and a sanitization oracle $\mathcal{O}^{\mathsf{no\text{-}Frame}}_{\mathsf{San}}$. The $\mathcal{O}_{\mathsf{Sign}}$ oracle is shown in Figure 4.8, while the three others are shown in Figure 4.17a.

Anonymity versus unlinkability. We highlight the fact that, although conceptually close, the properties of unlinkability and anonymity capture independent attack scenarios. In unlinkability, the adversary tries to link signatures modified for a single known sanitizer, while in anonymity, the adversary has to guess the identity of an unknown

$\mathcal{O}^{\text{no-Frame}}_{\text{Register}}$

1 : $(\mathsf{spk}, \mathsf{ssk}) \xleftarrow{\$} \mathsf{SaKeyGen}_{\text{k-SAN}}(1^\lambda)$
2 : $\mathcal{U} \leftarrow \mathcal{U} \cup \{(\mathsf{spk}, \mathsf{ssk}, 1)\}$
3 : **return** spk

$\mathcal{O}^{\text{no-Frame}}_{\text{del}}(\mathsf{sk}, \underline{\mathsf{spk}, l \leq k})$

1 : $\mathsf{del} \leftarrow \mathsf{Delegate}_{\text{k-SAN}}(\mathsf{sk}, \mathsf{spk}, l)$
2 : $\mathcal{D}[\mathsf{spk}] \leftarrow (\mathsf{del}, l)$
3 : $\mathcal{H}[\mathsf{spk}] \leftarrow \emptyset$
4 : **return** del

$\mathcal{O}^{\text{no-Frame}}_{\text{San}}(\mathsf{pk}, \underline{\mathsf{spk}, m, \sigma, \mathsf{MOD}, \eta})$

1 : Extract $(\mathsf{spk}, \mathsf{ssk}, b)$ from $\mathcal{U}$
2 : **if** $b = 0 \vee \mathcal{D}[\mathsf{spk}] = \perp \vee \eta \in \mathcal{H}[\mathsf{spk}]$:
3 : **return** $\perp$
4 : $\mathcal{D}[\mathsf{spk}] \xrightarrow{p} (\mathsf{del}, l)$
5 : **if** $\eta > l$: **return** $\perp$
6 : $\sigma \leftarrow \mathsf{Sanitize}_{\text{k-SAN}}(m, \sigma, \mathsf{MOD}, \mathsf{ssk}, \mathsf{pk}, \mathsf{del}, \eta)$
7 : $\mathcal{H}[\mathsf{spk}] \leftarrow \mathcal{H}[\mathsf{spk}] \cup \{\eta\}$
8 : **return** σ

(a) Description of the $\mathcal{O}^{\text{no-Frame}}_{\text{Register}}$, $\mathcal{O}^{\text{no-Frame}}_{\text{del}}$ and $\mathcal{O}^{\text{no-Frame}}_{\text{San}}$ Oracles for Non-frameability of k-SAN.

$\mathsf{Exp}^{noFrame}_{\mathcal{A},\text{k-SAN}}(1^\lambda, n)$

1 : $\mathcal{U}, \mathcal{D}, \mathcal{H} \leftarrow \emptyset$
2 : $(\mathsf{pk}, \mathsf{sk}) \xleftarrow{\$} \mathsf{Gen}_{\text{k-SAN}}(1^\lambda, k, n)$
3 : $(m_i^*, \sigma_i^*)_{i=1}^2 \leftarrow \mathcal{A}^{\mathcal{O}^{\text{no-Frame}}_{\text{Register}}, \mathcal{O}^{\text{no-Frame}}_{\text{del}}, \mathcal{O}^{\text{ano/no-Frame}}_{\text{Sign}}, \mathcal{O}^{\text{no-Frame}}_{\text{San}}}(\mathsf{pk})$
4 : $(\mathsf{id}, w) \leftarrow \mathsf{Link}_{\text{k-SAN}}(\mathsf{pk}, m_1^*, \sigma_1^*, m_2^*, \sigma_2^*)$
5 : **if** $\exists$ssk s.t. $(\mathsf{id}, \mathsf{ssk}, 1) \in \mathcal{U}$: **return** 1
6 : **return** 0

(b) Description of the Non-frameability Experiment.

Figure 4.17: Description of Non-frameability Experiment and Oracles for k-SAN.

sanitizer for a given message and can control the modifications this sanitizer makes to these signatures.

Since in the anonymity game the adversary chooses for itself how and by whom signatures are modified via oracles, and cannot have a signature modified by several different sanitizers, knowing how to link a sanitized signatures to its original gives it no advantage. Note that for signatures sanitized by the unknown sanitizer that the adversary has to determine, the sanitization oracle will always use the key of the unknown sanitizer, thus avoiding trivial attacks where the adversary tests whether the sanitization of its signature by a chosen sanitizer fails or not.

On the other hand, for the unlinkability experiment, the adversary receives a signature sanitized by a given user, and must determine the original signature key used. As the original signature can only be sanitized by one sanitizer chosen *a priori* by the signer, guessing the identity of this sanitizer by attacking anonymity gives the adversary no advantage. So there is no implication between unlinkability and anonymity.

Note that when the limit of k sanitizations is exceeded, it is the identity of the sanitizer and the link between their signatures that is leaked, but it is still not possible to link the sanitized signatures to the original signatures. We link the signatures of a sanitizer, but the unlinkability property still holds for these signatures.

4.4.2 k-Times Anonymous Sanitizable Signature Scheme

Our k-times anonymous sanitizable signatures combine the design of the sanitizable signatures in [BLL+19, BB21] with the mecanism we introduced in our k-times anonymous proxy signature. The signature contains commitments that allow the sanitizer to show that only admissible blocks are modified. More precisely, the sanitizer provides a proof that for every block within the altered message, the commitment corresponds to the hash of the index or the hash of the index combined with its content. If any unauthorised block is altered, then the sanitizer is unable to generate the proof. In addition, the sanitizer produces elements that enable our tracing mechanism to work if it exceeds its sanitization limit. In order to achieve transparency, we show how the signer can simulate these elements in the original signature. This results in two computationally identically distributed signatures outputed by $\mathsf{Sign}_{\mathsf{San}}$ and $\mathsf{Sanitize}_{\mathsf{San}}$. In what follows, we describe our k-SAN scheme.

The Setup algorithm is the same as in Section 4.3.2.

$\underline{\mathsf{Gen}_{\mathsf{k\text{-}SAN}}(1^\lambda, k, n)\colon}$ if $n > 1$, sets $l = \lceil \log_2(k) \rceil$, and generates two SPS keys pairs $(\mathsf{pk}^{\mathsf{del}}_{\mathsf{SPS}}, \mathsf{sk}^{\mathsf{del}}_{\mathsf{SPS}}) \leftarrow \mathsf{Gen}_{\mathsf{SPS}}(1^\lambda, 4l+1)$ and $(\mathsf{pk}^{\mathsf{MOD}}_{\mathsf{S}}, \mathsf{sk}^{\mathsf{MOD}}_{\mathsf{S}}) \leftarrow \mathsf{Gen}_{\mathsf{SPS}}(1^\lambda, 2n)$. The the algorithm samples $\mathsf{sk}_{\log} \xleftarrow{\$} \mathbb{Z}_p^*$ and set $\mathsf{pk}_{\log} = g_1^{\mathsf{sk}_{\log}}$, and returns $\mathsf{pk} = (\mathsf{pk}^{\mathsf{del}}_{\mathsf{SPS}}, \mathsf{pk}^{\mathsf{MOD}}_{\mathsf{SPS}}, \mathsf{pk}_{\log})$ and $\mathsf{sk} = (\mathsf{sk}^{\mathsf{del}}_{\mathsf{SPS}}, \mathsf{sk}^{\mathsf{MOD}}_{\mathsf{SPS}}, \mathsf{sk}_{\log})$.

$\underline{\mathsf{SaKeyGen}_{\mathsf{k\text{-}SAN}}(1^\lambda)\colon}$ samples $\mathsf{ssk}_{\log} \xleftarrow{\$} \mathbb{Z}_q$, set $\mathsf{spk}_{\log} = g_1^{\mathsf{ssk}_{\log}}$, runs $(\mathsf{ssk}_e, \mathsf{spk}_e) \leftarrow \mathsf{Gen}_{\mathsf{Enc}}(1^\lambda)$ and returns $\mathsf{ssk} = (\mathsf{ssk}_{\log}, \mathsf{ssk}_e)$ as the secret key and $\mathsf{spk} = (\mathsf{spk}_{\log}, \mathsf{spk}_e)$ as the public key.

$\underline{\mathsf{Delegate}_{\mathsf{k\text{-}SAN}}(\mathsf{sk}, \mathsf{spk}, k)\colon}$ sets $l = \lceil \log_2(k) \rceil$, abort if the SPS key $\mathsf{sk}^{\mathsf{del}}_{\mathsf{SPS}}$ does not support messages of $4l+1$ group elements. For all $(i,j) \in [\![l]\!] \times \{0,1\}$, samples $x_{i,j} \xleftarrow{\$} \mathbb{Z}_p^*$, set $y_{i,j} = g_1^{x_{i,j}}$, $\mathsf{spk}_{i,j} = \mathsf{spk}_{\log}{}^{x_{i,j}}$ and producesx the SPS signature $\widehat{\sigma} \leftarrow \mathsf{Sign}_{\mathsf{SPS}}(\mathsf{sk}^{\mathsf{del}}_{\mathsf{SPS}}, (g_1, y_{1,0}, \ldots, \mathsf{spk}_{l,1}))$. Returns $\mathsf{del} = ((x_{i,j}, y_{i,j}, \mathsf{spk}_{i,j})_{i \in [\![l]\!]; j \in \{0,1\}}, \widehat{\sigma})$.

Below, we describe the $\mathsf{Sign}_{\mathsf{k\text{-}SAN}}$ and $\mathsf{Sanitize}_{\mathsf{k\text{-}SAN}}$ algorithms, drawing parallels between their similarities and specifying their respective executions when they differ.

Both $\mathsf{Sign}_{\mathsf{k\text{-}SAN}}(m, \mathsf{ADM}, \mathsf{sk}, \mathsf{spk})$ and $\mathsf{Sanitize}_{\mathsf{k\text{-}SAN}}(m, \sigma, \mathsf{MOD}, \mathsf{ssk}, \mathsf{pk}, \mathsf{del}, \eta)$ sets $l = \lceil \log_2(k) \rceil$. Then:

$\underline{\mathsf{Sign}_{\mathsf{k\text{-}SAN}}}$**:** Parses $m \xrightarrow{p} m_1 \| \dots \| m_n$, samples $\eta \xleftarrow{\$} [\![0, k-1]\!]$, $s \xleftarrow{\$} \mathbb{Z}_p^*$, and $\widehat{g}_1, \widehat{y}_{i,j}, \widehat{\mathsf{spk}}_{i,j} \xleftarrow{\$} \mathbb{G}_1$ for all $(i,j) \in [\![l]\!] \times \{0,1\}$. Simulates a delegation by signing $\widehat{\sigma} \leftarrow \mathsf{Sign}_{\mathsf{SPS}}(\mathsf{sk}_{\mathsf{SPS}}^{\mathsf{del}}, (\widehat{g}_1, \widehat{y}_{1,0}, \dots, \widehat{\mathsf{spk}}_{l,1}))$. For all $i \in [\![l]\!]$ and $j \in \{0,1\}$, sets $\widetilde{y}_i = \widehat{y}_{i,\eta[i]}^{s}$, and $\widetilde{\mathsf{spk}}_i = \widehat{\mathsf{spk}}_{i,\eta[i]}^{s}$.

$\underline{\mathsf{Sanitize}_{\mathsf{k\text{-}SAN}}}$**:** Parses $\mathsf{MOD}(m) \xrightarrow{p} m_1 \| \dots \| m_n$, $\sigma \xrightarrow{p} (\mathsf{del}_\sigma, \mathsf{tra}, \pi_\sigma)$, $\mathsf{del}_\sigma \xrightarrow{p} (\widehat{g_1}, \widetilde{g_1}, \{\{\widehat{y}_{i,b}, \widehat{\mathsf{spk}}_{i,b}\}_{b\in\{0,1\}}, \widetilde{y}_i, \widetilde{\mathsf{spk}}_i\}_{i\in[\![l]\!]}, \widehat{\sigma}, \Pi_{<k})$ and $\mathsf{tra} \xrightarrow{p} (\{u_i, v_i\}_{i=1}^n, \widetilde{y}, \alpha_1, \alpha_2, \alpha_3, \alpha_4, \tau, \pi_{\mathsf{MOD}}, \sigma_{\mathsf{MOD}}, e)$ (note that the values of most of these variables will be updated by reallocation during the algorithm). Then the algorithm proceeds similarly to the initial steps of the Sign algorithm of the k-APS signature scheme, halting before the execution of the proof $\Pi_{<k}$.

Both algorithms generate the proof $\Pi_{<k}$ of knowledge of s and η which proves that (i) $\widetilde{y}_i$ and $\widetilde{\mathsf{spk}}_i$ are well formed according to s and some integer η of l bits and (ii) $\eta < k$. This proof follows the same instantiation as before. To conclude this first part, set $\mathsf{del}_\sigma = (\widehat{g_1}, \widetilde{g_1}, \{\{\widehat{y}_{i,b}, \widehat{\mathsf{spk}}_{i,b}\}_{b\in\{0,1\}}, \widetilde{y}_i, \widetilde{\mathsf{spk}}_i\}_{i\in[\![l]\!]}, \widehat{\sigma}, \Pi_{<k})$.

Both algorithms start the second phase by setting the message blocks:

$\underline{\mathsf{Sign}_{\mathsf{k\text{-}SAN}}}$**:** To mandate the sanitizer for a set of modifiable blocks: samples $a \xleftarrow{\$} \mathbb{Z}_p^*$. For all $i \in \mathsf{ADM}$ let $u_i = H(m_i, i, 0)^a$ and $v_i = H(m_i, i, 1)^a$, otherwise let $u_i = H(i,0)^a$ and $v_i = H(i,1)^a$. Encrypt $e \leftarrow \mathsf{Enc}(\mathsf{spk}_e, a)$

$\underline{\mathsf{Sanitize}_{\mathsf{k\text{-}SAN}}}$**:** Samples $b \xleftarrow{\$} \mathbb{Z}_p^*$, decrypt $a \leftarrow \mathsf{Dec}(\mathsf{ssk}_e, e)$ and updates $e \leftarrow \mathsf{Enc}(\mathsf{spk}_e, a \cdot b)$. Sets $\mathsf{ADM} = \emptyset$ and $\forall i \in [\![n]\!]$, let $u_i = H(m_i, i, 0)^{a\cdot b}$ and $v_i = H(m_i, i, 1)^{a\cdot b}$ when the signature contains $H(m_i, i, 0)^a$ and $H(m_i, i, 1)^a$, otherwise let $u_i = H(i,0)^{a\cdot b}$ and $v_i = H(i,1)^{a\cdot b}$. Checks if $\mathsf{MOD} \subset \mathsf{ADM}$ and then sets $a = a \cdot b$, otherwise return $\perp$.

Both algorithms generate the signature of knowledge:

$$\pi_{\mathsf{MOD}} \leftarrow \mathsf{SoK}_e \left\{ a : \bigwedge_{1\le i\le n} \begin{array}{l} (u_i = H(m_i,i,0)^a \wedge v_i = H(m_i,i,1)^a) \\ \vee\, (u_i = H(i,0)^a \wedge v_i = H(i,1)^a) \end{array} \right\}.$$

$\underline{\mathsf{Sign}_{\mathsf{k\text{-}SAN}}}$**:** executes $\sigma_{\mathsf{MOD}} \leftarrow \mathsf{Sign}_{\mathsf{SPS}}(\mathsf{sk}_{\mathsf{SPS}}^{\mathsf{MOD}}, (u_1, v_1, \dots, u_n, v_n))$. Sets $\widetilde{y} = \prod_{i=1}^{l} \widetilde{y}_i$, $\widetilde{\mathsf{spk}} = \prod_{i=1}^{l} \widetilde{\mathsf{spk}}_i$, $u = \sum_{i=1}^{n} u_i$, $v = \sum_{i=1}^{n} v_i$ and samples the elements $\alpha_1, \alpha_3, \alpha_4 \xleftarrow{\$} \mathbb{G}_1$, $\alpha_2 \xleftarrow{\$} \mathbb{G}_2$ and a tracing element $\tau \xleftarrow{\$} \mathbb{G}_t$.

$\underline{\mathsf{Sanitize}_{\mathsf{k\text{-}SAN}}}$**:** Adapts σ_{MOD} given randomness b: $\sigma_{\mathsf{MOD}} \leftarrow \mathsf{ChgRep}_{\mathsf{SPS}}((u_1, v_1, \dots, u_n, v_n), \sigma_{\mathsf{MOD}}, b, \mathsf{pk}_{\mathsf{SPS}}^{\mathsf{MOD}})$. Set $x = \sum_{i=1}^{l} x_{i,\eta[i]}$, $\widetilde{y} = \prod_{i=1}^{l} \widetilde{y}_i$, $\widetilde{\mathsf{spk}} = \widetilde{y}^{\mathsf{ssk}_{\log}}$, and computes $\alpha_1 = h_1^x$. Let $u = \sum_{i=1}^{n} u_i$, $v = \sum_{i=1}^{n} v_i$ and samples $t \xleftarrow{\$} \mathbb{Z}_p^*$. Computes $\alpha_2 = g_2^t$, the matching elements $\alpha_3 = h_2^x \cdot g_1^{u\cdot \mathsf{ssk}_{\log}}$, $\alpha_4 = h_3^x \cdot h_4^{v\cdot \mathsf{ssk}_{\log}}$ and a tracing element $\tau = e(h_4, \alpha_2)^{\mathsf{ssk}_{\log}}$.

The vector of elements $\mathsf{tra} = (\{u_i, v_i\}_{i=1}^n, \widetilde{y}, \alpha_1, \alpha_2, \alpha_3, \alpha_4, \tau, \pi_{\mathsf{MOD}}, \sigma_{\mathsf{MOD}}, e)$ is set by both entities and embed in a signature of knowledge where the sanitizer proves the first

part of the `or` statement and the signer the second part:

$$\pi_\sigma \leftarrow \mathsf{SoK}_{(\mathsf{del},\mathsf{tra})}\{\mathsf{sk}_{\log}\colon (\widetilde{y} = \widehat{g}_1^{\,x\cdot s} \wedge \widetilde{\mathsf{spk}} = \widetilde{y}^{\mathsf{ssk}_{\log}} \wedge \alpha_1 = h_1^x \wedge \alpha_2 = g_2^t \wedge$$
$$\alpha_3 = h_2^x \cdot g_1^{u\cdot \mathsf{ssk}_{\log}} \wedge \alpha_4 = h_3^x \cdot h_4^{v\cdot \mathsf{ssk}_{\log}} \wedge \tau = e(h_4,\alpha_2)^{\mathsf{ssk}_{\log}}) \vee (\mathsf{pk}_{\log} = g_1^{\mathsf{sk}_{\log}})\}.$$

Finally $\mathsf{Sign}_{\mathsf{k\text{-}SAN}}$ and $\mathsf{Sanitize}_{\mathsf{k\text{-}SAN}}$ return the signature $\sigma = (\mathsf{del}_\sigma, \mathsf{tra}, \pi_\sigma)$.

Signature verification consists of re-computing the elements that are necessary for the verification of every SPS and signature of knowledge. The linking and tracing algorithms are the same as $\mathsf{Link}_{\mathsf{k\text{-}APS}}$ and $\mathsf{Trace}_{\mathsf{k\text{-}APS}}$.

We will now informally recall the security properties of k-SAN and explain why they hold for our scheme, except for the anonymity, traceability and non-frameability that are reached in a similar way as in our k-APS scheme (Section 4.3.2).

Unforgeability. The users cannot generate a valid signature without knowing a secret key which has obtained a delegation. This property relies on the hardness of recovering the secret key of the signer or one of the sanitizers, which is ensured by the DDH assumption. Once this is ruled out, we can reduce the ability of an adversary to forge a signature to its ability to forge SPS signatures.

Immutability. A sanitizable signature is *immutable* when no adversary is able to sanitize with unauthorised modification. This property relies on the collision resistance of the hash function, as well as the soundness and zero-knowledge properties of the signature of knowledge π_{MOD} (as they link the message to the signature). Moreover, the EUF-CMA security of the SPS and the DDH assumption prevent impersonation of the signer.

Transparency. The verifier cannot decide whether a given signature has been sanitized or not, which means that the outputs of $\mathsf{Sign}_{\mathsf{k\text{-}SAN}}$ and $\mathsf{Sanitize}_{\mathsf{k\text{-}SAN}}$ should be computationally indistinguishable. The randomised delegation encompassed in the signature is identically distributed to a newly produced one. All SoKs can be produced by both the signer and the sanitizer, while the other elements are shown to be computationally indistinguishable based on the DDH problem.

Invisibility. The invisibility property prevents an adversary which is neither the signer, nor the sanitizer of a signature from determining any information on the modifiable blocks. The difference between a modifiable block and a non-modifiable block is the input of the hash function serving as a commitment. The obtained hash is then elevated to a secret random power. Therefore, invisibility mainly relies on the class hiding property (Definition 10 on page 31).

Unlinkability. For a fixed sanitizer assigned with two signatures, the verifier cannot link a sanitized signature with its original version. In the proposed signature scheme, all elements undergo randomization during sanitization or are entirely new, which ensures this property.

We therefore have the following theorem, for which the proofs are available in [BOA24b].

Theorem 2: Security of our k-SAN.

When instantiated by a signature on equivalense classes that is unforgeable, class-hiding, and signature adaptatable, by NIZK proofs which are zero-knowledge and sound, by a collision-resistante hash function, by an SoK that has perfect simulability and simulation-extractability, and by an IND-CCA public key encryption, our k-SAN is unforgeable, immutable, transparent, unlinkable, anonymity, invisible, k-traceable and non-frameable under the DDH assumption in $\mathbb{G}_1$ and $\mathbb{G}_2$ in the random oracle model.

4.5 Design Variants

In this section, we show how to adapt our techniques to a variety of situations, by presenting minor variants of our schemes.

k-times group signatures. As mentioned in the introduction, an anonymous proxy signature scheme can be generically transformed into a group signature scheme, considering the delegator as a group manager and the delegates as group members. The idea remains valid for k-times signatures, so our scheme can be viewed as the first fully k-times group signature.

Full anonymity for proxy signature We considered that anonymity was not desirable from the point of view of the original signer. Indeed, as delegates sign on behalf of the original signer, it seems legitimate for it to know their identity through the proxy/sanitized signature. However, in some cases, we might want to guarantee full anonymity for the delegates, especially if we use our scheme as a group signature, as it is explained in the Introduction. Anonymity is not guaranteed because the secret keys $x_{i,j}$ are chosen by (and therefore known) the delegator, which can therefore, from a signature, find the keys $x_{i,j}$ used to forge x and thus trace the delegate's identity from $\widetilde{y}$. To prevent this, the delegate simply has to choose the $x_{i,j}$'s secretly. For the delegator, delegation therefore consists in signing public keys whose secrets it does not know. Without this information, as these public keys are randomized with a random element known only to the delegate, the delegator will no longer be able to lift the anonymity of the signatures under the DDH assumption. In particular, being the author of the equivalence class signature is of no help to the delegator, since once randomized, this signature is perfectly indistinguishable from a new signature, even for the user which generated it.

k-times anonymity versus k-times unlinkability/transparency In this chapter, we have focused on sanitizable signatures where the sanitizer is anonymous, the identity of the sanitizer being revealed after the k-th signature. However, if the delegate is not anonymous, the k-times approach can be applied to other information protection properties, in particular unlinkability and transparency.

The k-times unlinkability ensures unlinkability as long as the signature has not been sanitized more than k times. Once the limit is exceeded, all signatures derived from that signature can be linked to each other (note that the limit applies to a single signature, so the other signatures will remain unlinked). Our scheme can be easily adapted for this property: for each signature, the original signer uses the Delegate algorithm, and adds the del element to the e cipher. The sanitizer then recovers del by decrypting e, and can use it to sanitize the signature. Note that in this paradigm, there is no need to communicate a key to the sanitizer *a priori*. Signature size remains logarithmic in k.

If we consider sanitizable signatures linkable by design, we can use the k-times limit for transparency: if the limit is exceeded, it is possible to distinguish the signatures produced by the sanitizer from the original, which guarantees an accountability mechanism that is triggered without the active participation of the signer. This property can be ensured by the same mechanism as for k-times unlinkability. This variant is similar to the scheme in [CJ10], except that the scheme in [CJ10] leaks the sanitizer's secret key, which seems less reasonable for practical use as discussed in the introduction.

4.6 Conclusion of the Chapter

In this chapter, we have studied how delegation can be modified and restricted to a given number of k signatures when necessary to limit the proxies' capabilities for practical or legal reasons. We believe that the application of our method, involving a new zero-knowledge proof, can be easily applied to many real-world contexts. We defined k-times full traceable anonymity for proxy signatures and sanitizable signatures. In both cases, we define a security model, give an efficient scheme (in the sense that the size of keys and signatures is logarithmic in k), and prove its security. We have ended this chapter by describing minor variants of our schemes which can easily be derived from the two introduced concepts.

Outside the scope of our work, there are still limitations that if removed make the usage of these signature easier or more secure. In all our constructons, the use of SPS implies two limitations: the need for the generic group model, and the need to set an upper bound for k (the size of vectors signed by SPS must be prespecified). Proposing a construction without SPS would simplify our schema and provide more security guarantees for our constructions.

Another limitation we would like to fix in the future is the use of a fixed upper bound on parameter generation for k and for the number n of blocks in the sanitizable signature. This limitation is inherent in the use of SPS, as the size of vectors signed by SPS must be prespecified.

Chapter 5
EMV-compliant and Usable Anonymity for Contactless Payments

Chapter Summary

EMV (Europay Mastercard Visa) is the worldwide, de facto card-payment system we all use. Despite modern facilities, *in-shop* EMV contactless payments are not anonymous or private. The payers' long-term identification data leaks to Merchants or even to observers. Furthermore, the bank that issued the card are also capable to potentially profile their account-holders. The same set of laws and regulations that protects users against fraud lead to this lack of privacy. Balancing these regulations, we propose two privacy-enhancing, EMV-compatible contactless payment protocols: *PrivBank* and *PrivProxy*. For these proposals we:

- Define desired privacy properties: payers' anonymity regarding various entities, merchant anonymity concerning payers' banks, and payment unlinkability.
- Detail the feasibility of our solutions and their compliance with EMV payment regulations (AML, KYC, SCA).
- Compare our proposals with traditional payment solutions, evaluating their alignment with AML, KYC, and SCA regulations.
- Formally prove these properties for our proposals and compare them with other payment solutions.

Contents

5.1 Introduction

EMVCo [emv22a] is the largest consortium of payment providers, including the ubiquitous Visa, Mastercard, American Express, and UnionPay. In this chapter, we are interested in *EMV card-present transactions*, as this is indeed still the largest payments market, representing 94% of the entire payment market [EMV23a], with 12.8 billion bankcards in circulation. Yet, in the last 10 years, the EMV standard has substantilly evolved, adding numerous new features primarily to support contactless and mobile payments at scale. The security of even the newest EMV-protocols has been scrutinised by many [AT17, BCI20, BCDD20]; unfortunately however the privacy of the entities involved, and especially payers, has not.

No Privacy in EMV by Default. Indeed, despite its modern features, EMV payments do not provied privacy guarantees: in-shop, contactless EMV payments, be it by "plastic"/physical card or mobile device, allow for payees and payers to be tracked. For starters, payment providers (*e.g.,* Visa, Mastercard) link together mobile and plastic-card payment data, in their standard form, for instance, in the context of loyalty schemes and statistics [Inc24]. In addition, card-issuing banks as well as EMV payment networks always know where we shop, with which merchants, at which location, at what time, and can potentially use this data to profile us. Worse, despite the fact that all modern, EMV-compliant mobile-apps such as SamsungPay, GooglePay, and ApplePay generate and use a new "account number" with every payment one makes, merchants can link any of our purchases together, even if we make some with GooglePay, others with ApplePay,

and other, with SamsungPay. What is more, there is no complexity in creating these links: just observing the in-shop transaction between the payment device (*i.e.,* card or phone) and the merchant's payment-terminal/PoS (Point of Sale) suffices. In fairness, some customers may have signed up to a loyalty scheme with such a store, yet these customers may be unaware that their in-shop/by-PoS transactions, made with standard payments devices (payment cards, mobile-phone payment apps), can be linked together, even without the use of their loyalty card.

A Growing Interest in Anonymous Payments. While some users may not be concerned with this loss of privacy, others would like to be able to remain anonymous if possible [Han11, SLZ20]. Clearly, this is not possible for all EMV contactless purchases (*e.g.,* subscription-based ones), but in some cases there is no reason why it should not be possible, as was previously the case with banknotes.

Existent Privacy-aware EMV: Online Only and Potential for Improvements. Certain banks (such as Revolut [Rev23]) have taken steps towards better EMV-payment security and with that they gained some (weak) form of pseudonymity by emiting cards for one-time use. However, these cannot be used for in-shop/by-PoS purchases, but rather only for online shopping. Moreover, such solutions lack a degree of end-to-end usability: a one-time card is generated on an app, then the payer has to go online separately and pay with it.

Meanwhile, anonymous payments have been considered since the introduction of electronic payments in the 1980 in the form of e-cash [Cha83b, Raz02], and large-scale projects such as GNU TALER [BDGS16, Dol19] aim to revive that. Their drawback (with respect to our goals) is that they are not EMV-compliant, and in fact they would need to replace in-shop EMV transactions with a new *online* payment system. Deploying that at a large scale would be costly.

Adding Usable Privacy to EMV In-shop Contactless Payments. In conclusion, there is no in-shop/by-PoS contactless payment solution existis that provides pseudonymity and unlinkability for payers and for merchants, whilst being EMV-compliant, or as usable as normal EMV contactless payments.

Indeed, achieving this is not trivial: due to money laundering and fraud-protection regulations, legal requirements inherently do not align with privacy-preservation in EMV contactless in-shop payments, making it hard to attain. In other words, pseudonymity in EMV is hindered by the need for EMV payments to be auditable by the decision-making entities. There are several *Anti-Money Laundering* (AML) regulations [EU21b], including *Know Your Customer (KYC)*; Furthermore *Strong Customer Authentication (SCA)* [EU18] regulations to protect customers from fraud. There are also further fraud-protection mechanisms which are not mandatory, but lack thereof increases the risk of financial losses for banks and financial bodies, as they have to reimburse customers for unauthorised use of their cards. So, there is one added complexity to proposing a scheme for privacy-preserving EMV contactless, in-shop payments: making that scheme abide by the laws and regulations governing EMV.

Contributions. Given the above state below our concrete contibutions, and set the tone of this chapter.

Our first contribution is to create a in-shop, contactless payments with some degree of anonymity, which remain EMV-compliant from the viewpoint of system-requirements' engineering. Our goal is to render this payment method usable.

It is clear given the above, that the solution must not add heavy privacy-enhancing cryptographic machinery (such as homomorphic encryption, *etc.*) on top of EMV payments. Such solution would immediately lead to a system that is not compliant with today's EMV back-ends, but will also likely increase the duration of a payment, likely infringing the timing constraints that are in place today (*i.e.,* a contactless payment end-to-end takes only 12 seconds [Tak24]). Thus, our payment scheme will closely follow existing EMV payments, adding to it not privacy-enhancing cryptography but rather privacy-enhancing parties, such as anonymising proxies and instant escrows. We pursue this idea in Section 5.10 on page 131.

Since the first layer of our contribution relies on adding privacy-enhancing proxies to the infrastructure, we discuss how these proxies fit in with pseudonymity-hindering AML, SCA laws, and fraud-prevention mechanisms.

At present time, these pseudonymity-hindering AML and SCA laws and fraud prevention mechanisms are primarily observed and implemented by card-issuing banks. With this in mind, there are two avenues forward for payment-anonymising proxies in EMV: (a) the issuing banks provision these proxies themselves ; (b) third-party proxies are used, introducing a trade-off between liability and risks between the third-parties and the issuing banks.

Here, we provide two designs, *PrivBank* and *PrivProxy*, which can be plugged directly into the banking system. They modulate liability and risks differently, as per (a) and (b) above. We compare them, including in terms of usability, see Section 5.10.5 on page 139.

We subsequently formulate adequate pseudonymity requirements of payment schemes in a way that is clear and easy to understand by lay persons, since they will need to choose the levels of privacy and products suited to them. Moreover, these easy-to-understand privacy/pseudonymity properties need to be general enough to fit not just EMV, but also other payment systems that use similar parties, so that lay persons can compare and contrast. We do just this in Section 5.5 on page 120 (pseudonymity notions), Section 5.6 on page 122 (assessing other payment means against these notions) and Section 5.11 on page 140 (analysis for `PrivBank` and `PrivProxy`).

This intuitive and usable model for privacy and pseudonymity is subsequently compared to more traditional cryptographic formalisms for privacy and pseudonymity. Indeed, to gain confidence in a privacy-reasoning model such as the one presented in Section 5.11 on page 140 (which is based on building relations between data), one may need to see how it translates to a well-accepted model, *e.g.,* the cryptographic game-based one. We do this in Section 5.12 on page 146, giving cryptographic game-based definitions of payment anonymity, proving them equal to the intuitive ones given beforehand. We prove that our two designs `PrivBank` and `PrivProxy` attain these formal cryptographic definitions in the game-hoping proofs, as well.

Thus, in this chapter, we will balance various engineering, usability, security, and legal aspects to address the above and create solutions for in-shop, by-PoS contactless payments, which provably provide pseudonymity and unlinkability for payers and for merchants, whilst being EMV-compliant, arguably law-abiding and usable in the current infrastructure.

5.2 Acronyms

In this chapter, we frequently use various acronyms to streamline the presentation of complex terms and concepts. For the reader's convenience, we provide a list of these acronyms along with their full forms below. Readers may choose to skip this section, as the acronyms are all recalled or already defined in the text.

EMV	Europay Mastercard Visa
EMV-L	EMV language
EUR	Euro
EU	Europe
UK	United Kingdom
GBP	Great Britain Pound
AML	Anti-Money Laundering
KYC	Know Your Customer
SCA	Strong Customer Authentication
PSD2	Payment Services Directive (version 2)
MCC	Merchant Category Code
MN	Merchant Name
MRI	Merchant Risk Index
ML	Merchant Location
PAN	Application Primary Account Number
PAR	Payment Account Reference
TEE	Trusted Execution Environment
CVM	Cardholder Verification Method
CDCVM	Consumer Device CVM
CBDC	Central Bank Digital Currency
PSP	Payment Service Provider
T&C	Terms and Conditions
CBDC	Central Bank Digital Currency

Table 5.1: List of Acronys Used in this Chapter.

5.3 Related Work

In Sections 5.5 and Section 5.6 we describe the privacy allowed by payments schemes with respect to our interests, that is law-abiding, regulation compliant, and privacy-provisioning. We also compare our solution to the privacy-preserving architectures introduced in this chapter.

Research into EMV is vast, ranging from applied works such as [GY19, Yun21, TY21] to formal treatments [RCN+22, MDAB10, DRP12, BCM+14, AT17, BCI20, BSTP21, BCDD20]. Most formalisms for EMV analysis are based on the symbolic/Dolev-Yao model [DY83], very few are computational (*e.g.,* [BSWW13]) like the one we gave in

Section 5.13. Also, all these formalisms look at security, not at privacy like us. We are the first to give a model based on mathematical relations (see Section 7) to encode privacy in EMV. The closest idea to this, not in EMV, speaks of traceability relations [BBW+23], but, these are complex links made between protocol layers. Next, we will cover works related to ours directly: on privacy or close to traditional payment systems.

EMV Payments and Privacy. [BCM+14] showed that long-term, plastic-cards' PANs can be used to track people. This gave rise to tokenised, mobile-devices' EMV PANs, which would go towards $\mathcal{P}\texttt{An}$, should it not be for their associated introduction of PARs in EMV. Later, mobile EMV payments, including tokenisation were studied, *e.g.,* [AT17], [CFF+17], but from a security perspective, not a privacy one. Some aspects therein, even if not privacy-centric, are relevant to us: the requirements we make of our app not be corrupted are grounded on certain mobile wallets having been shown [AT17] to be it.

In our case, the card-to-PoS channel is insecure as per current EMV specifications. Meanwhile, [HMY22, BHMY23] work in the setting of future-generation EMV, where this channel will be secure. In this setting, considering a corruption model and linkability attack stronger than one against `Unlnk`, [HMY22] find future-generation. EMV payments to be linkable. The authors of [BHMY23] build on [HMY22], extend their model to dishonest terminals and achieve unlinkability and anonymity for smart card-based payments. Both proposals yielded non-EMV compliant patches.

EMV payment tokenisation for mobile payment has now been standardised [EMV21]. Althrough our work heavily relies on EMV, its usage raises new questions, particularly in terms of the trust that can be placed in the smartphone hosting the application. *Trusted Execution Elements* which are *Secure elements* are part of the solution. Their progresive deployment on smartphones raises new questions on how to use them efficiently. To answer this question, Cortier *et al.* [CFF+17] has introduced an EMV-compliant tokenisation system that makes practical use of the secure element.

The authors of [AT17] took a closer look at the security aspects of mobile payments. In their work, they took into account the threats to privacy and the various adverse attacks that mobile systems have twart while implementing EMV compatible payments. They also examine how NFC-enabled mobile wallets exchange sensitive transaction data with contactless point-of-sale terminals. These considerations are deemed necessary to implement our proposals (see the discussion in Section 5.10.5).

Other EMV or Traditional Payments. Cryptocurrencies [NB08] are alien to EMV. But, non-EMV payments close to EMV exist. For instance, Lyf [Lyf23] and Visa [Vis24] propose payment services which rely on their own payment network and QR codes. A large-scale, EU-funded project tries to push new payments based on the GNU Taler initiative [BDGS16] and using the well-known e-cash idea by Chaum [Cha83b]. They perform online transactions and are compliant with online-payments' regulation, they do not use the card-to-PoS-merchant payment networks like us. Attaining privacy via online transactions is easier – *e.g.,* via one-time cards without the worry of "in-shop" SCA but relying on 3D secure, without having to share credentials over an app between different-domain entities. Further in this paper, we do not consider these payment methods because they differ from the payment network already in place.

5.4 A Preamble to Our Solution

We take inspiration from existing payment systems: plastic-card and mobile EMV, disposable EMV cards, proxying of EMV payments by Curve [Cur23], and machinations during EMV-payment authorisation.

(A) On payers' Pseudonymity and Payments' Unlinkability. The main inspiration for our designs here come from mobile EMV-payments and one-time/disposable cards for online shopping, and we bring the latter into the space of "in-shop" payments. As a result of enhanced security, mobile payments are already more privacy-preserving than plastic cards, as they hide the main identifier of the physical card, the *Primary Account Number* (PAN), via ephemeral card-like number called *tokenised PAN*. This enhances towards payers' pseudonymity. Yet, mobile-payments made via the same bankcard still contain a fixed card-identifying value called the *Payment Account Reference (PAR)*. Which enables the linkability of payments. All our designs will revert to tokenisation and PAN-PAR-based constructions in mobile payments. Instead, our mobile apps use one-time disposable cards which produce transactions as per plastic cards, which is akin to having a one-time PAN. This does not impact security but significantly enhances user anonymity.

(B) On merchants' Pseudonymity. Here, we take inspiration from EMV-payment proxies such as Curve [Cur23] (see Section 5.7 on page 127). We add an intermediary party in the interaction between the payer and the merchant, which also relays the payment to the issuing bank, stripped of certain merchant-related data. In more detail, based on an agreement between the issuer and the proxy with respect to, *e.g.,* certain categories of merchants with sufficiently low *Merchant Risk Indicators*, the proxy omits sending the merchant name to the issuer, while still providing the latter with some merchant identification data.

However, there is one last hurdle to our designs, chiefly the sets of regulations, as follows.

(1) AML and counter-terrorist financing regulations require the auditability of payments by certain payment-system parties; therefore, for any transaction, the payer and the merchant must be traceable.

(2) SCA/(*Payment Services Directive*) (PSD2) require identification of payers prior to using a payment service, including opening bank-accounts and making payments.

Therefore, we need to carefully combine the ideas in (A) and (B) above to achieve payers and merchant pseudonymity as well as payment unlinkability, and, simultaneously EMV-compliance and abiding by regulations (1) and (2) above.

To achieve this, some entities may retain some of the identity information required and, if all combined, the system can be controlled in accordance with laws (1) and (2).

5.5 Payments-Privacy Notions

We propose privacy notions from the perspective of different entities in the payment systems: a *payer* who pays for goods/services, a *merchant* selling them, an *issuer* who gives the payer a means of payments (bank account, cards, banknotes, *etc.*), and a *proxy* who mediates the purchase from the payer to the merchant.

5.5.1 Entities Identification in EMV

Law-Enforced Payer Identifications. Firstly, in EMV payments, the payer has to be identified at the on-boarding phase with its banks. To obtain a bank card, customers must provide a piece of identification such as a passport and proof of address to the card-issuer. These measures fall under the *Know Your Customer (KYC)* regulations. Secondly, in EMV payments, the payers have to be identified during transactions: its PIN (Personal Identification Number), biometrics, similar knowledge or possession must be checked as they pay. This is know as *Strong Customer Authentication (SCA)* and it is governed by Payments Security Directive (PSD2) regulations [EU21a]. Thirdly, payers must be identified (even beyond SCA) for all transfers or payments amounting to certain values over a certain period (*e.g.,* EUR/GBP1000 per month in the EU/UK). Together, it is all driven by fraud protection (*e.g.,* in the case of SCA) and/or *AML includes The Money Laundering and Terrorist Financing Regulations (AML)* regulations [EU21b, Actb, Acta]. The Financial Conduct Authority (FCA)'s Handbook FCG 3.2.5 [Aut] requires regulated firms such as issuers to perform (real-time) *monitoring* of transactions and submit "Suspicious Activity Reports" (containing the payer identity with the payment details) to the FCA, if concern arise.

Merchant Identification. In EMV payment systems, the issuers usually get the following merchant information with each of its customers' transactions: *Merchant Category Code (MCC)*, *merchant's name*, *Merchant Risk Index (MRI)*, *Merchant Location (ML)*.

Issuer Identification. The identity of the issuer is generally disclosed to the entities participating to a payment, and of no interest to outer parties. If there is a proxy in the system, this proxy may learn who someone banks with, but one can argue that most proxies would be chosen, and are less of a mainstream worry therefore. Symmetrically, if a party is proxying payments, its pseudonymity is less crucial than that of the payer and the merchant.

We have formalised all this in the threat model of Section 5.11 on page 140.

5.5.2 Our Payment-Privacy Notions

Now, we are in a position to put forward a set of desirable privacy notions, generic enough that they can be meaningful in most payment systems. These properties can be acticulated from the perspective of the different relevant entities involved: with respect to a merchant, an issuer, a proxy or an observer.

Payer Pseudonymity ($\mathcal{P}$An). An instance of $\mathcal{P}$An holds if a given entity does not get to know a payer's identity ID or a long-term pseudonym.

The $\mathcal{P}$An property comes in various flavours with respect to what is termed as identifiable "object". For instance, in EMV-like payment systems:

$\mathcal{P}\text{An}^{ID}$**:** imposes that the merchant does not learn the payer's long-term identity ID;

$\mathcal{P}\text{An}^{C_{ID}}$**:** imposes that the merchant does not learn the payer's long-term card-number or bank account.

By contrast, in EMV payments even made by mobile phone, the merchant does get to learn always a long-term pseudonym of the payer. Although there is generally no implication between $\mathcal{P}\text{An}^{ID}$ and $\mathcal{P}\text{An}^{C_{ID}}$, as demonstrated below, our constructions realise both at the same time.

$\mathcal{P}\text{An}^{C_{ID}} \not\Rightarrow \mathcal{P}\text{An}^{ID}$**.** For instance, paying for goods with the long-term card currently reveals the PAN/PAR of the card C_{ID} to the (PoS of the) merchant, but the identity of the payer ID is not disclosed in the payment.

$\mathcal{P}\text{An}^{ID} \not\Rightarrow \mathcal{P}\text{An}^{C_{ID}}$**.** A payer utilising a payment-proxy system (*e.g.,* Revolut, see Section 5.6 on the next page) may undergo a KYC procedure with the proxy thus leaking its ID, but this said proxy may never know a long-term card identifier C_{ID} of the payer, as the latter may always top up its proxy account from one-time cards.

Payments' Unlinkability (Unlnk). An instance of Unlnk holds if a relevant entity in payment systems will remain unable to link payments made by the same payer.

For instance, if one pays by cash twice in the same store, the banknotes used are not per se linkable to this same payer. Thus, a merchant should ideally be unable of knowing if two EMV payments are made by the same payer; unfortunately, this is not the case with current EMV-payments.

Merchant/Seller Pseudonymity ($\mathcal{M}$An). An instance of $\mathcal{M}$An holds if an entity in a payment system (*e.g.,* an issuer) does not infer the identity of the merchant involved in a payment. During a regular EMV transaction, merchant-related information is transmitted to the acquirers, making it possible to identify the merchant and to monitor the risks of potential fraud. It is not clear which data is required to validate a transaction and which could remain optional. We note that some data such as the *merchant category code* does not necessarily allow full identification of the merchant. Other metadata, such as the *Merchant Identifier* (which is an acquirer-dependent identifier), gives full knowledge of the merchants identity. Given the need for fraud management, it is worth mitigating full merchant pseudonymity for practical use and toward improving the acceptance rate. Therefore, we also restricted the set of merchants with similar characteristics (*e.g.,* ML, MCC).

Discussions On $\mathcal{P}$An, Unlnk, and $\mathcal{M}$An. As expected, $\mathcal{P}$An and Unlnk are considered from the viewpoint of the merchant or an observer, since the issuer may always know

Payment method	SCA		KYC		Pseudonymity		
	Issuer	Proxy	Issuer	Proxy	Issuer	Merchant	Proxy
1. Cash	no	N.A.	N.A.	N.A.	$\mathcal{M}$An	$\mathcal{P}$An, Unlnk	N.A.
2. Cheque	y	N.A.	y	N.A.	$\neg\mathcal{M}$An	$\neg\mathcal{P}$An, $\neg$Unlnk	N.A.
3. E-cash	y	N.A.	y	N.A.	$\mathcal{M}$An	$\mathcal{P}$An, Unlnk	N.A.
4. Physical cards	y	N.A.	y	N.A.	$\neg\mathcal{M}$An	$\neg\mathcal{P}$An, $\neg$Unlnk	N.A.
5. Google, Apple Pay, *etc.*	y	N.A.	y	N.A.	$\neg\mathcal{M}$An	$\mathcal{P}$An, $\neg$Unlnk	N.A.
6a. Top-up cards	y	N.A.	y	N.A.	$\neg\mathcal{M}$An	$\mathcal{P}$An/$\neg\mathcal{P}$An, $\neg$Unlnk	$\neg\mathcal{P}$An, $\neg$Unlnk, $\neg\mathcal{M}$An
6b. Pre-Paid/gifts cards	no	(n)	n	(n)	$\mathcal{M}$An/$\neg\mathcal{M}$An	$\mathcal{P}$An, Unlnk/$\neg$Unlnk	($\neg\mathcal{P}$An, $\neg$Unlnk, $\neg\mathcal{M}$An)
7. Virtual cards	y	(y)	y	(y)	$\mathcal{M}$An/$\neg\mathcal{M}$An	$\mathcal{P}$An, Unlnk	($\neg\mathcal{P}$An, $\neg$Unlnk, $\neg\mathcal{M}$An)
8a. PayPal	y	y/no	y	y/no	$\mathcal{M}$An	$\mathcal{P}$An, $\neg$Unlnk	$\neg\mathcal{P}$An, $\neg$Unlnk, $\neg\mathcal{M}$An
8b. Curve	y	y	y	y	$\neg\mathcal{M}$An	$\neg\mathcal{P}$An, $\neg$Unlnk	$\neg\mathcal{P}$An, $\neg$Unlnk, $\neg\mathcal{M}$An
9. Online Marketplaces	y	no	y	y/no	$\mathcal{M}$An	$\mathcal{P}$An, $\neg$Unlnk	$\neg\mathcal{P}$An, $\neg$Unlnk, $\neg\mathcal{M}$An
PrivBank	y	no	y	no	$\mathcal{M}$An	$\mathcal{P}$An, Unlnk	$\mathcal{P}$An, $\neg$Unlnk, $\neg\mathcal{M}$An
PrivProxy	y	y	no	y	$\mathcal{M}$An	$\mathcal{P}$An, Unlnk	$\neg\mathcal{P}$An, $\neg$Unlnk, $\neg\mathcal{M}$An

Table 5.2: SCA, KYC and pseudonymity properties of payment methods from the point of view of the Issuer, Merchant and the Proxy when it exists. $\neg\mathcal{P}$An and $\neg$Unlnk holds in all systemes for the Issuer and $\neg\mathcal{M}$An holds in all systems for the Merchant. Detailed explanations are provided in the Section 5.6. *(Notation: N.A. stands for "not applicable", $\mathcal{P}$An for $\mathcal{P}\mathtt{An}^{ID}$ and $\mathcal{P}\mathtt{An}^{C_{ID}}$, brackets are used when the proxy may not necessarily exist in all systems and / when deployment can lead to different properties.)*

the payer's identity due to KYC. Similarly, $\mathcal{M}$An its most naturally required from the issuer's viewpoint.

Additional relations between these properties apply: unlinkability is required for $\mathcal{P}$An, *i.e.,* $\neg\mathcal{P}$An $\Rightarrow$ $\neg$Unlnk. Indeed, an entity deducing the identity of the payer or information that is persistent across transactions can link said transactions. However, the reverse does not hold, *i.e.,* $\mathcal{P}$An $\not\Rightarrow$ Unlnk. An example for the above is provided by EMV payment tokenisation [EMV22c]: there, each payment yields an ephemeral identifier for the card/payer called *tokenised Primary Account Number (PAN)*: *i.e.,* $\mathcal{P}\mathtt{An}^{C_{ID}}$ holds. But payments by the same payer are linkable together by a data called *Payment Account Reference (PAR)*: *i.e.,* Unlnk does not hold. The merchant and any observer between the payer and the merchant PoS get these tokenised PANs and PARs.

5.6 Traditional Payment Systems and Their Privacy

We now look at what one may call *"traditional" payment systems* and some *modern* ones[1], yet built on the previous ones, and discuss where they sit with respect to KYC and SCA regulations, and how they fare against our privacy requirements $\mathcal{P}$An, Unlnk, and $\mathcal{M}$An. We exclude crypto currencies [NB08] and QR-code-based payments [Lyf23]. The reason for this is that their infrastructure is totally different from other long-established payments, especially the EMV-based systems, which we aim to augment here. Thus, the systems of interest in our analysis are: 1. cash, 2. cheque, 3. e-cash [Cha83b], 4. physical/classical bankcards, 5. mobile-phone apps [App23, Pay23], 6. top-up and pre-paid cards [Inc23, Vis22], 7. virtual/one-time cards [Rev23], 8. payment service providers such as 8a. PayPal [Pay22] and 8b. Curve [Cur23] and 9. online marketplaces [Ama23, eBa23]. We believe the reader may be familiar with most of these systems to the level where it can judge if they have SCA, KYC, $\mathcal{P}$An, Unlnk, $\mathcal{M}$An.

Analysis of KYC, $\mathcal{P}$An, Unlnk, $\mathcal{M}$An in Payments. Table 5.2 shows the way payment systems fare against SCA and KYC requirements (as described in Section 5.5.1

[1]Online payments are included here as they may give some insights into what might be possible.

on page 120), and our notions $\mathcal{P}\texttt{An}$, $\texttt{Unlnk}$, $\mathcal{M}\texttt{An}$ (see Section 5.5.2 on page 120). We obtained these results via an empirical analysis, as they are straightforward[2] and do not require formal argument. For instance, e-cash, physical card, Payment Apps (Google Pay, Apple Pay, *etc.*), top-up and virtual cards are all payments card for which the SCA/KYC apply to the issuer, unlike pre-paid cards, which fall under both exemptions cases. PayPal applies strict limits unless the customer is identified, and thus the application of rules depends on the usage scenario. For Online Marketplace, we assume that they act as an intermediary between the purchaser and the merchant, and, no KYC or SCA is deemed required in their case. In this section, we look at them and consider aspects linked to our main interests. We provide a description of each of them and give reasons for the claimed properties.

1. Cash (banknotes and coins). This works for in-shop purchases only. There is no KYC and strong pseudonymity, although shops may use security cameras and subsequent review may allow the payer to be identified, this is beyond the scope of the payment method. Banknotes might be traced, based on their serial numbers, but that requires a complicated set of steps taken by various banks and the Merchant, which again turns the matter beyond into a complex type of identifiability. Coins do not feature serial numbers. Thus, we can say that cash has no KYC and provides pseudonymity and unlinkability (within reasonable/normal measures). For the same reason, namely that cash cannot be linked to the payer, SCA cannot be required before payments.

2. Cheques. They provide a payment mechanism which involves banks as an intermediary between the payer and the merchant or the entity that is being paid. In addition, cheques are required to indicate the name of the payer and of the merchant, as well as a bank account number for the payer. The account number also divulges the bank issuing the cheque to the payer. This method therefore has KYC, but provides no pseudonymity. In many countries, it has become common practice to require a proof of identity in order to make a payment by cheque. In such cases, the payer is strongly authenticated and SCA holds.

3. E-cash. This method was originally proposed by Chaum in his 1983 paper on "Blind Signatures for Untraceable Payments" [Cha83b]. Payers have an e-wallet topped-up with e-cash, from which they spend e-cash like they would real cash. Limits on the expenditure exist though, *e.g.,* for AML reasons. Thus, this digital mechanism aims to provide the user with the same level of pseudonymity that they achieve when using cash. If the link between the e-cash wallet and the owner's bank account is just for applying limits to the amount of e-cash stored on a wallet, then the merchant should not obtain information about the customer and even the bank should gain little information. How it is regulated and implemented will determine the outcomes which may differ from Table 5.2 on the preceding page.

We proceed to give further details on e-cash, since it approches our goals and methods. E-cash requires a separate network to EMV, but it still has supporters today. Chaum commercialised this idea, by founding DigiCash in 1994. This and other early

[2]The main value of this analysis is reviewing/systematising these systems and concepts.

e-cash developments are described in [Raz02]. Although none of these have been commercially successful, there is still interest in developing e-cash systems [BDGS16].

In the US, the "Electronic Currency And Secure Hardware Act" [eca22] proposes that an electronic dollar should be created with the same privacy properties as the dollar itself. If achieved, there would be no *Static Data Authentication* (SDA) requirement and this could also be used for online purchases, although for full pseudonymity the purchaser may need to use a VPN to hide its IP address, use a separate e-mail account and a delivery locker for its purchase.

In contrast, the European Central Bank's document discussing e-cash (Central Bank Digital Currency, CBDC) [Ban21] recommends to avoid too much money being stored in consumer's digital wallets, and not in the banks. Hence limits are recommanded and e-cash payments are enabled, if the consumer is identified and their CBDC wallet is linked to their bank account. Others [gnu22] opposed it, and it is not clear what will ultimately be decided.

4. Plastic/physical credit or debit cards. Such cards are used in shops or online to make payments in ways we are all familiar with. We have deferred the description of payment cards to the Section 5.7.1 on page 127. In the the meantime, we focus on the statement provided in Table 5.2 on page 122 with respect to such payments KYC, SCA, and pseudonymity.

5. Google Pay [Pay23] and Apple Pay [App23]. These are two of the most used methods of mobile payment, *i.e.,* payment via a mobile app "inside" which a physical card is registered. Like in physical cards, KYC and SCA are the norm here. In mobile-payment SCA, when making the payment, the customer may be asked to confirm their identity onto the payment device too, via PIN, fingerprint, or face recognition. But, unlike payments by physical cards, making and authorising mobile payments require both intermediaries in the payment networks, and tokenisation (as outlined in Section 5.7.1 on page 127). Due to this tokenisation mechanism, the merchant can link purchases by the same payer using their long-term PAR – created once during the onboarding of their card onto the app and used in all their payments thereafter. The PAR replaces the PAN, which is then hidden from the merchant, making payments anonymous without unlinkability.

6. Pre-paid cards. These are cards which may have no bank-account associated to them and one tops him up with a set amount or one buys reading topped up and uses. We divide pre-paid cards into two categories:

a) Top-up cards – cards offered with services such as those by Revolut [Rev23]. Other providers exist: for example, UK pre-paid Mastercard cards providers, listed on the Mastercard website, for general use and as a gift all behave as per the below. The Payer may not have to undergo credit checks, but must satisfy identity and address checks and have money available in their account to cover any payments made. So, KYC is generally done. Indeed, most of the issuers of such pre-paid cards act as electronic money institutions and are regulated by the Electronic Money Regulations [leg11]. Since these are cards, their pseudonymity properties

are the same as those of plastic cards, or that of mobile apps – if they are loaded therein.

b) Gift cards – card that can be bought is store cards, with set amounts preloaded onto them. There is generally no KYC. Their use is thus restricted (to specific merchants) and the amount on each card is small, to satisfy AML requirements [EU21b]. If they are purchased with cash and not linked to a bank account (for re-charging, for example), then, subject to the same caveats about IP addresses, e-mail accounts, and deliveries, payer pseudonymity can be achieved.

7. Virtual or "one time" cards (VC). These provide pseudorandom card details (card number, expiry date, CVV), for each transaction. A payer can remain anonymous to the merchant, but the virtual card is linked to that 'real' card to enable payment to be made and so the issuer knows who the payer is and from whom they are purchasing. These are marketed for online use, in general. One example is the Revolut card [Rev23]. Another company, Swidch [Swi23], offers a range of services based on One Time Access Codes (OTAC) and this includes *ephemeral* cards. As for other virtual cards the latter are linked to a registered *real* payment card (requiring KYC). In this context, an intermediary may be acting as a payment proxy, although how individual providers handle the payments differs from one company to another. In addition, unless strict usage limits are applied, SCA is required for all payments.

8. Payment service providers. There are a number of providers in this category, and we list a few other examples below.

8a. PayPal. PayPal offers a range of services. In terms of the discussion here, PayPal accounts can be used to make payments. Figure 5.1 shows how PayPal is used for this purpose. The stages are:

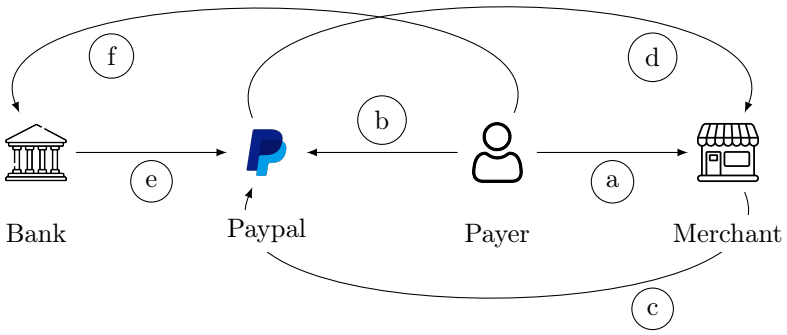


Figure 5.1: Making payments with PayPal.

1. The cardholder opens an account and registers a card to be used for payment. SCA for the Issuer is carried out at this point, but is not necessary afterwards. Unless the account holder confirms their identity and address (KYC), accounts are restricted and have limits placed on amounts that can be sent, received, or

withdrawn [Pay22]. PayPal is acting as the *payment service provider* (PSP) and knows who the payer is and gets to see who is making the sale and when.

2. The cardholder purchases an item from a merchant and pays using their PayPal account. Purchases can be made online or in person.

3. The merchant receives their payment *from PayPal* dnd not directly from the payer. Thus the purchaser may use a pseudonyme and separate the email account and a delivery locker for their purchase in order to obtain full pseudonymity.

4. PayPal pays the merchant.

5. PayPal charges the issuer. The issuer knows that something was purchased, but not from whom.

6. The cardholder pays the issuer.

In this case, payment or identity information has been provided to PayPal, but has not been made known to the merchant because PayPal has filtered most of it. Other information, such as email, or perhaps also pseudonyms, is not linked to the payer's identity.

Amazon Pay. Amazon Pay [Ama23] offers a similar service to PayPal and allows paying online with a credit card, debit card or by direct debit. They make it clear that the merchant does not receive the payer's payment details: "We do not share your full credit card, debit card, or bank account number with sites or charitable organizations that accept Amazon Pay. The merchant only receives information that is required to complete and support your transaction. This information may include your name, email address, and shipping address".

8b. Curve. Curve [Cur23] provides a payer with a card and a payment application. Curve users must satisfy KYC rules. The payer registers multiple bankcards issued by one or several banks in the Curve app.

When the payer pays using their Curve card or the Curve app, the payment network (*e.g.,* Visa, Mastercard) and the bank authorise the payment based on Curve card data and merchant information, as the Curve card interacts with the merchant's PoS system and both sends their own data. Curve pays the merchant on behalf of the payer on the spot, and one of the payer's Curve-registered bankcards is charged, but the payment is not entirely "settled" (*i.e.,* the payment onto the classical card remains "pending").

Curve provides the payer a period of up to 120 days to potentially move the transaction to another of their registered bankcards. This results in an intricate "payment authorisation" process involving Curve, the traditional bankcard issuers, and the payment network. During this process, all necessary information is shared between the entities, so that each party knows the identities involved.

9. Online Marketplaces. Examples of these are provided by Amazon and eBay. We view these are merchants here. So, from that viewpoint, clearly, here is generally no KYC or SCA needed to open accounts with them, as a payer for their goods. However,

eBay, for example, states in their terms [eBa23] that they may require "any other data about the buyer which the buyer's payment service provider or we may require". Aside from that, if goods bought from them are sent to a pickup locations, then some degree of pseudonymity can be achieved.

5.7 Our Main EMV Ingredients

We recall the notions related strictly to EMV payments which are relevant to us focusing on aspects linked to Payers' and Merchants'[3] identification and related regulations.

5.7.1 From Card Issuing to Payment Processing

We divide EMV card-based payments in 4 main stages also represented in Figure 5.2.

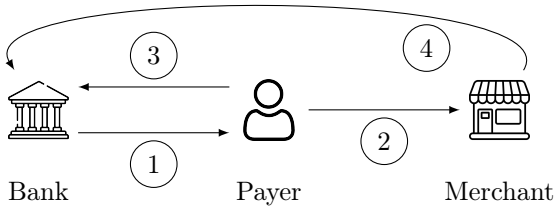


Figure 5.2: Overview of typical card payments.

1. Card-issuing, KYC and AML. A future Payer opens a bank-account with a *card-issuer* (*i.e.,* bank). We discuss the case where it receives a credit/debit card, associated to the account. The card is supplied by one of the current *card providers* (e.g,. Mastercard, Visa, American Express, ...). We will losely refer to the collection of Issuers, card providers and the proxies linking them as *payment networks*. To obtain such a card, the customers must provide a piece of identification, such a passport and proof of address to the card-issuer, in line with KYC regulations. KYC falls under the AML regulations [EU21b] (see Section 5.5.1 on page 120).

2. Making a Card Payment and Relevant Payment Data: PAN, MCC, ML. A *cardholder* goes to pay with their card to a *Merchant*, using their card-readers, known also as *points of sale*: the cardholder inserts the card into the PoS or taps the PoS in the case of contactless transactions. The basic operations of the protocol between the card and the PoS are defined in the set of standards from EMVCo. A partial example of the data exchanged between the card and the reader, called the *payment transcript*, is available in Section 5.8 on page 129.

Due to the transmission of the PAN, plastic-card EMV does not have $\mathcal{P}\texttt{An}$ from the viewpoint of the Merchant, or an observer between the card and the Merchant's PoS.

At the end of the protocol, certain transaction data is signed by the card and returned to the PoS for its checks. Equally, the card sends a MAC of certain transaction data to

[3]Capital letters are used to refer formally to entities in the system: *e.g.,* "payer" – a personal paying, in Sections 1-4, vs "Payer" – a formal algorithmic party, from Section 5 onwards .

the PoS to forward it to the card-issuers for checks therein. They check this data and based on it, they approve/decline the transaction.

Alongside the card-centric data sent on the back-end from the Merchant's PoS to the Issuer, payment networks and others, the Merchant's PoS also adds some or all of the following merchant-identifying details, relevant to us: *Merchant category code (MCC)*, *Merchant's name (MN)*, *Merchant risk index (MRI)*, *Merchant location (ML)*. Thus, due to the transmission of the MN, plastic-card EMV does not have $\mathcal{M}\texttt{An}$ from the viewpoint of the Issuer.

3. Customer Identification During Payments. When the card is presented to the Merchant's PoS for payment, the SCA/PSD2 [EU18, EU21a] regulation requires two factor authentication of the Payer (*e.g.,* possession of card and associated PIN). The Issuer checks the payment data sent by the Merchant along with this SCA identification-data of the payer. Should the checks fail, the payment is declined by the bank. There are variations to card and PIN verifications, especially if the payment is not made by card[4]. If the payment is contactless, derogation to the SCA rules can apply and single-factor authentication is required instead. SCA remains required every few payments or after a set of payments has exceeded a spent value (*e.g.,* EUR/GBP150 in EU/UK).

4. Payment Authorisation and Clearing. Funds are settled during the final phase, called *clearing*, as follows. (i) The Merchant, via its acquirer, requests payment from the card issuer. The issuer verifies details like transaction location, the payer's identity, and merchant information. (ii) If the cardholder has sufficient funds, the issuer deducts the amount from its account[5]. The final authorization is handled by the issuing bank, possibly in consultation with payment networks like Visa or Mastercard. Once approved, the funds are transferred to the Merchant's/acquiring bank.

As stated in Section 5.4 on page 119, the information necessary for a payment authorisation varies based on the business model (*e.g.,* from Visa to Mastercard) and not all Merchant information is truly necessary. For example, Curve [Cur23] operates (and *e.g.,* Visa incentivise it [Vis21]): they over-submit Merchant data especially if its MRI is high, so as to increase the probability of authorisation and therefore maintain customer satisfaction. There may be leeways in provisioning $\mathcal{M}\texttt{An}$ from the viewpoint of the Issuer, since the minimal amount of Merchant-data needed proportionate to its partial role in payment authorisation is not standardised.

5.7.2 Mobile Payments: Tokenisation and Transaction Data

Mobile payment applications such as ApplePay [App23], Google Pay [Pay23] *etc.* allow plastic cards to be used via a mobile application. The onboarding requires an authorisation from the card's issuing bank, and therefore KYC is observed. The Payer can then use the app for contactless payments. When a payment is made, the card's long-term PAN is *tokenised*, and the payment transcript between the phone and the Merchants PoS

[4]E.g., if the Payer uses a smartphone SCA verification by the issuing bank is replaced by *Consumer Device Cardholder Verification Method* (CDCVM) executed on the phone. That is, the payers fingerprint or face identification is read by the phone and used as customer authentication. The result of that is later checked by the issuing bank.

[5]This is for debit cards. For credit cards, this differs slightly.

looks different from one made with the physical card, with PAN-related data replaced by tokenised values. Mobile-payment transcripts (see Section 5.9 on the following page for an example) include the following payer-identifying data relevant to us:

- *one-time tokenised PAN* – an ephemeral account number that changes with each payment and each app: each payments made with card C through mobile app A_1 or app A_2 will each generate a different number.
- *long-term PAR* – a fixed value that is shared amongst various/all payment(s) apps $A_1, A_2 \ldots$ to refer to *any/all* payment made based on the same pysical card C. The PAR value was introduced at the request of the Merchants and payment networks, so that one card C used for mobile payments hence, showing varying tokenised PAN, can all be linked together.

The tokenised PAN and PAR values are sent by the Merchant onto the payments networks, just as the "plastic" PAN was. However, before these reach the Issuer, the tokenised PAN is de-tokenised by entities in the EMV system which transform it back to the associated PAN. The rest of the backend part of payment processing is as described in Section 5.7.1 on page 127. Details of payment tokenisation and use cases are given in [EMV22c, EMV23b].

From the viewpoint of the Merchant, mobile EMV payments achieve a form of $\mathcal{P}\texttt{An}$, via the PAN, but do not achieve `Unlnk`, due to the PAR. Now, we can re-state our aim: *augment mobile EMV payments to obtain* $\mathcal{P}\texttt{An}$, $\texttt{Unlnk}$, *and* $\mathcal{M}\texttt{An}$ *without breaking AML, KYC and PSD2/SCA .*

5.8 Sample Real Card Traces

Here we give a number of traces from card-based payments, with the goal of illustrating the points made earlier about the data included in card transactions. Here is part of a trace from a MasterCard card:

```
  5A | len:8  Application Primary Account Number: 5521573039705376
5F24 | len:3  Application Expiration Date YYMMDD: 240430
5F25 | len:3  Application Effective Date YYMMDD: 200401
5F28 | len:2  Issuer Country Code: 0826
5F34 | len:1  Application Primary Account Number Sequence Number: 01
```

The card record contains the PAN and the expiration date. We can also find information determining the currency in which the card is issued. Note: In the same trace we also see:

```
9F02 | len:6  Amount, Authorised (Numeric): 000000004600
9F03 | len:6  Amount, Other (Numeric): 000000000000
9F1A | len:2  Terminal Country Code: 0826
  95 | len:5  Terminal Verification Results: 0000008001
5F2A | len:2  Transaction Currency Code: 0826
  9A | len:3  Transaction Date: 210318
```

```
9F35 | len:1  Terminal Type: 22
9F34 | len:3  Cardholder Verification Method Results: 1F0302
1F No CVM required;
```

This part shows the amount to be paid and other information about the terminal, such as the currency and the country where it is located. The cardholder verification method is also indicated. Merchant information comes from the terminal and are only sent to the acquirer in the backend, which is why it is not shown in the trace.
The same type of trace can be observed for other EMV-compatible card brands (Visa, AmericanExpress, *etc.*).

5.9 Sample Mobile Application Traces

Here we give a number of traces from mobile phone applications to illustrate the points made earlier concerning tokenisation and the PAR, starting with a part of a trace from an iPhone transaction:

```
  70 | len:37 Record Template
5F28 | len:2  Issuer Country Code: 0826
9F07 | len:2  Application Usage Control: C000
9F19 | len:6  Token Requestor ID: 040010030273
5F34 | len:1  Application Primary Account Number (PAN): 00
9F24 | len:29 Payment Account Reference (PAR):
              563030313030313330313631393633353535363931303530393738393 3
```

The Token Requestor ID and the Payment Account Reference are part of the tokenisation process.

In the same trace we also see:

```
70   | len:81 Record Template
5A   | len:8  Application Primary Account Number (PAN): 4831920272059474
5F24 | len:3  Application Expiration Date YYMMDD: 231231
9F46 | len:176  ICC Public Key Cert:
                149DD6A920995B05A5146C6ABEE823AFD2E2CBE91C701C2648E395EA
                D23F5AD04C8C6B2D2DA4CD271B154339C1AB342E683964F812CA7C67
                2E15F0407E6D3A9253E064F9ECD01A49DD7D4C5B22388367F9C26108
                FCC4AE2D94B169A322E29F65B02438FC0EC648AB949BAE006E270C4F
                17E52B40D11E2CDC8782C7AB873FD625119DB250AED39E3CFF60F526
                35708BB36ED60C8FEA5EC4
9F47 | len:1  ICC Public Key Expo: 03
```

However, the PAN in this case identifies the Payment Token and is not the primary account number (see EMV Payment Tokenisation Specification Technical Framework v2.2, Page 74). This is a "shared token".

A payment from a Samsung phone with the same card loaded into the payment application gives the following trace:

```
70   | len:81  Record Template
9F69 | len:7   Card Authentication Related Data: 01DC7BA93B0000
9F4B | len:128  Signed Dynamic Application Data (SDAD):
                80892716925C6DBD4AB0817A929D40A6D56DBE58535ACC74B05C491B
                AA28E62D6951FFC5F49DE9FAB97389DE800AFD04D391DEF44152C212
                F6959100B479BE204124F847A4C005481D3998EDD5C2349F50274900
                13DE56D77F98084EC49D748B2F45680075EF7F786813E6AF17851F26
                DFF92392363F85AEF8ED225F9E462C41
9F07 | len:2  Application Usage Control: C000
5F28 | len:2  Issuer Country Code: 0826
5F24 | len:3  Application Expiration Date YYMMDD: 231231
5A   | len:8  Application Primary Account Number (PAN): 4831920272190329
9F19 | len:6  Token Requestor ID: 040010043095
9F24 | len:29 Payment Account Reference (PAR):
              563030313030313330313631393633353535363931303530393738393 3
```

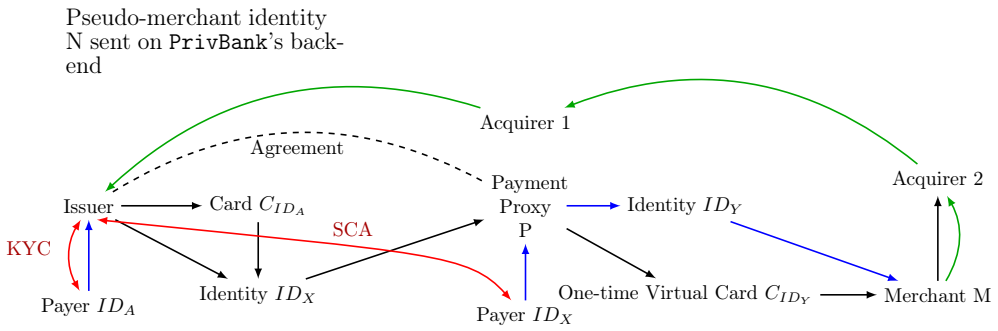
We notice that, while the Token Requestor ID and Payment Token are different, the PAR values are the same. According to the EMV FAQs on tokenisation [EMV22b] the PAR "was introduced to resolve the challenges faced in the broader acceptance community including Merchants, Acquirers and Payment Processors, in regards to linking Payment Token transactions with each other or transactions initiated on the underlying PAN. This supports a variety of payment processes and value added services". Thus, in this scenario the payments can be linked both by the Payment Token and more widely by the PAR.

5.10 Anonymous EMV In-Shop Payments

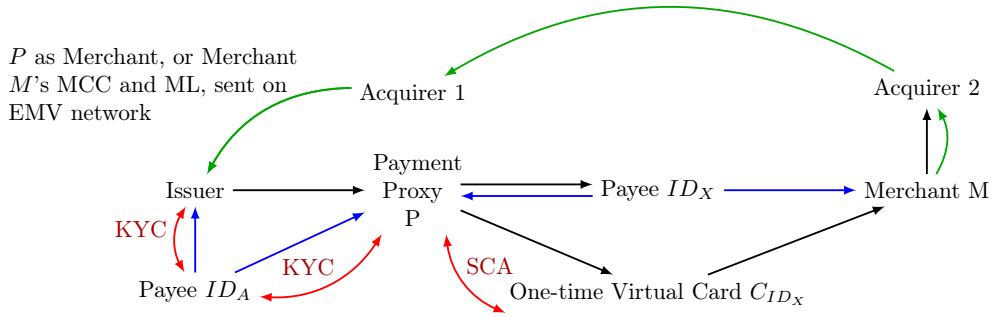
We proceed to propose two constructions, both compatible with EMV contactless payments, providing privacy as per $\mathcal{P}$`An`, $\mathcal{M}$`An`, `Unlnk`, with provable guarantees, all the while being compliant with the aforementioned laws and regulations governing these matters (SCA/PSD2, AML, *etc.*).

At the core of our first construction `PrivBank` depictied in two different representations in Figure 5.3a on the following page and 5.4 on page 133, there is a privacy-friendly, issuing bank which provisions $\mathcal{P}$`An` and `Unlnk` for its customers. To do this, this bank strongly partners with a Payment Proxy which mediates and curates customers' payments providing $\mathcal{M}$`An`. Meanwhile, at the heart of our second construction `PrivProxy`, depicted in Figure 5.3b on the following page and 5.5 on page 137, there is no longer a bank, but rather a pseudonymity-friendly Payment Proxy which aims to provide $\mathcal{P}$`An`, $\mathcal{M}$`An`, `Unlnk` of its own accord and at its own risks, to Payers who bank with whoever they chose to, independently of the Payment Proxy.

The crux of our designs is to compose several standard, non-private EMV-payments or parts thereof, such as to obtain one payment mobile, contactless EMV payment which is anonymous (w.r.t. $\mathcal{P}$`An`, $\mathcal{M}$`An`, `Unlnk`). We realise this via the design and use of proxies, without modifying EMV elements in the original payments, and without cryptographic additions. As such, all the cryptography used in our schemes and all



(a) `PrivBank`: EMV-Compliant Payments with Pseudonymity Provisioned Collaboratively by Privacy-friendly Issuer and Third-party Proxy



(b) `PrivProxy`: EMV-Compliant Payments with Pseudonymity Provisioned by Third-Party Proxy

Figure 5.3: Graphical description of the our two proposals. (Black arrows denote the execution flow. Red arrows are for KYC and SCA requirements. Blue arrows denote identity knowledge. Green arrows denote Clearing inquiries.)

EMV building blocks can be treated as black-boxes inherited from EMV, and our only focus is going to be the design of the proxied systems, from an engineering perspective alone. Indeed, our proofs w.r.t. $\mathcal{P}$`An`, $\mathcal{M}$`An`, `Unlnk` follow from the proxied construction, and the cryptographic or inner protocol details (*e.g.,* Visa, Mastercard variations) are irrelevant therein, as they are in the descriptions that follow.

We will now describe the functionality of our proposals by describing the main aspects and intricacies of `PrivBank` and `PrivProxy`.

5.10.1 Construction `PrivBank`

`PrivBank` (Figure 5.3a and 5.4 on the next page) can be summarised as follows. A Payment Proxy contractually committed with the bank, plays the intermediary between all in-store transactions done by a Payer with a Merchant. The Payment Proxy gets to know who the Merchants are, but not the long-term identifiers of the Payers, whereas the bank knows who the Payers are, but not the Merchants. As this is required by AML regulation, the two entities can work together to recover full knowledge on any transaction. They have an agreement in place, thus sharing risk and liability. It is important to note that an agreement can create civil liability *e.g.,* to indemnify the bank for a fine if the proxy fails to do something it has promised but it cannot apportion criminal liability or liability to pay a fine. Civil liability to the customer would only be

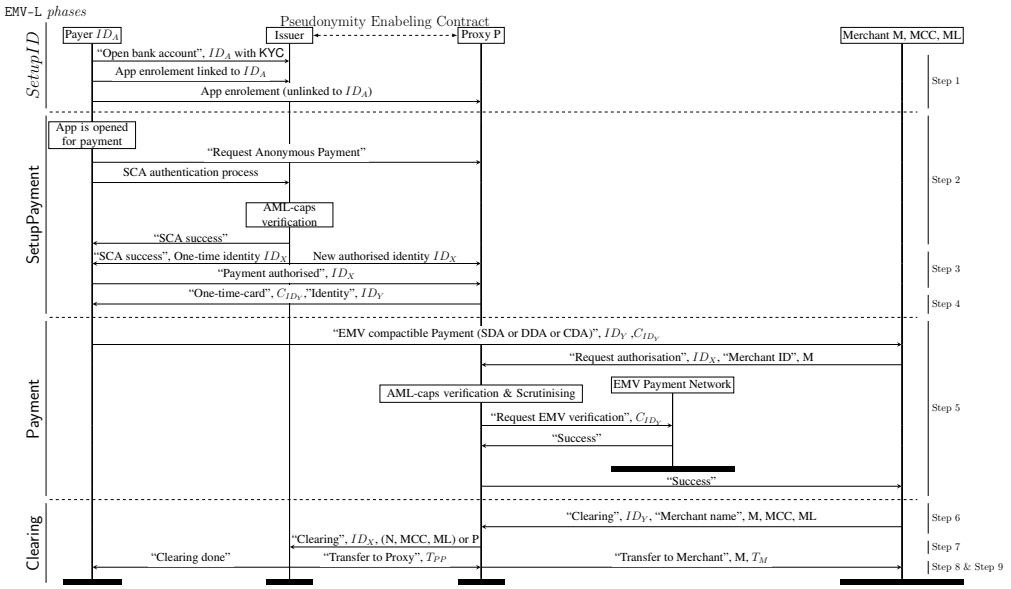


Figure 5.4: Protocol-Implementation Flow of `PrivBank`. *All communications appart from the payment from the Payer to the Merchant are assumed to be executed on a secure channel (encrypted and authenticated communications).*

by the bank. Payments done via `PrivBank` are supported via one mobile app[6] provided in partnership between, the Issuer and the Proxy. We now describe, step by step, how a payment is made possible as being well as carried out through via `PrivBank`. These steps are also highlighted in Figure 5.4.

Step 1. The Payer (whose identity is denoted by "ID_A" in Figure 5.3a on the preceding page and Figure 5.4) opens a bank account with the Issuer. This bank account comes with a "premium" option of support for privacy à la $\mathcal{P}$`An`, $\mathcal{M}$`An`, `Unlnk`, which is achieved via `PrivBank`. The Payer's banking and `PrivBank` 's accounts are with the Issuer, which handles KYC authentication, not the Proxy. The Issuer does not share the Payer 's identification details with the Proxy. To allow this, the contract stipulates certain terms and conditions (T&C) as we detail below.

As an account-holder with the Issuer, the Payer accesses the `PrivBank` app, which connects the Payer, Issuer, and Proxy from an engineering standpoint. However, the app is provided by the Issuer and links only the app-store identifier to the Payer's identity ID_A on the Issuer's servers. The Proxy and other third parties identify the Payer through their app-store account, not the Issuer's method. See *SetupID* on Figure 5.4.

Under AML regulations, and similar to other payment methods, the amounts spent using `PrivBank` may be capped (*e.g.,* EUR/GBP1000 per month). We call these the *AML-caps*. The T&Cs set this limit, which the Issuer and Proxy enforce.

Step 2. When a payment is to be made by the Payer to a PoS of a Merchant's, the Payer ID_A opens the `PrivBank` app. The opening of the app prompts both the Issuer and the Proxy on secure (*e.g.,* HTTPS) channels:

[6]For compliance with banking regulations, this app may require a Trusted Execution Environments (TEE), chips designed for secure storage and cryptographic operations.

(a) App-Triggers on the Proxy's Side: At this stage, the push by the app to the Proxy only says that someone, *with no specific identity revealed*, intends to make a payment.

(b) App-Triggers on the Issuer's Side: The SCA and AML checks are triggered through a request to the Issuer. SCA is a two-factor authentication, ensuring that Payer ID_A is making the payment. If needed, the Issuer also verifies that ID_A has not exceeded the AML caps for `PrivBank`. If the caps are reached or if SCA fails, the protocol halts.

Upon successful SCA and AML checks, the Issuer creates a one-time virtual identity ID_X, pseudorandom and statistically independent of ID_A/C_{ID_A}.

Step 3. On the back-end (*i.e.,* not via the `PrivBank` app), the Issuer sends the identifier ID_X to the Proxy, which, in the light of Steps 1 and 2, only knows that an account holder with the Issuer using `PrivBank` wants to make a payment.

The Proxy expects this push[7], having received an alert in Step 2 that someone intends to make a payment.

Step 4. At this stage, the Proxy issues – for a user it knows as Payer ID_X – a one-time, virtual, EMV-compliant card C_{ID_Y} with all aspects (PAN, *etc.*) being freshly generated for one-time use, included an attached one-time, virtual card-holder name of "ID_Y". ID_Y/C_{ID_Y} are pseudorandom and statistically independent of ID_X and of ID_A/C_{ID_A}. The card issued is "loaded" onto the app.

Step 5. The Payer pays with the one-time card by the Merchant's PoS. Here, SCA authentication may have to be carried out offline using the CDCVM tag in the case of preloaded payment methods ID_Y/C_{ID_Y}. The Proxy (and/or the Issuer) checks that the issuer would not reach the AML-caps through this specific amount being paid via `PrivBank` and executes the AML scrutinising. If any of these conditions fail, the protocol stops. AML caps were checked in Step 2 of `PrivBank`, but those checks did not account for the current payment.

Liability Shifts and Fraud-protection. Under its partnership with the Proxy, the Issuer accepts controlled *shift of liability* with respect to fraud protection. To this end, for *selected stores* – that are nominated based on MCC, MRI and ML, *etc.*– the partnership allows that the Issuer receives from the Proxy sanitized information when it comes to payment authorisation. In practice, the *list of selected stores* can be large (*e.g.,* all Merchants in a country with given MCCs), as is for the "*Ticket Restaurant*" services with Edenred [Ede23] or Up-one [UO23]. The sanitized information does not reveal the original Merchants' full identity, instead it contains so called *pseudo-merchant identities*. These are prescribed, such that the Issuer can check the Proxy's compliance to the agreement[8].

If fraud-detection disputes should be raised, the Proxy and the Issuer have to come together to resolve this, and the Proxy has to disclose to the Issuer the full Merchant data. This is reflected in the T&C of the contract that the Payer has with the Issuer on

[7]The time between any push by the app in Step 2 and receiving ID_X is capped at 2 seconds due to EMV security constraints.

using the PrivBank product, *i.e.,* the Payer knows that it can use PrivBank, in selected stores.

Step 6. Using standard EMV mechanisms, the (Acquirer of the) Merchant begins to resolve the payer's payment by contacting the Proxy, which is the Issuer of C_{ID_Y}.

Step 7. If the Merchant M is not on the "pre-selected" list, the protocol stops. Otherwise, using the PrivBank's back-end, the Proxy goes to Issuer, to resolve a payment for Payer ID_X, and provides a *pseudo-merchant identity N* instead of the true identity of the Merchant M.

Step 8. The Issuer checks if Payer ID_X can pay to a pseudo-merchant N via PrivBank as per the pre-agreed list of merchant and as per the rules of PrivBank. Then, further checks that Payer ID_X has funds to pay.

Step 9. If step 8 went through, the payment is resolved towards the Proxy and then from the Proxy to the Merchant.

5.10.2 Law Abiding and Norm Compliance Aspects of PrivBank

PrivBank comply with the norm and all regulation applicable to the banking system. We discuss why below.

On the Norm Compliance. Two aspects need to be examined: firstly, the payment transcript produced between the app and the Merchant's PoS. When a payment is made with this app, the transcription is that of the EMV contactless payment card, with the exception of the CDCVM, which can be filled in thanks to SCA verification with the Issuer.

Secondly, the timing compliance with the EMV norm. EMV-compliant payments are required to set a maximum general processing time for each transaction. Allowed timings range between a few hundred milliseconds to a few seconds. In the stages defined in PrivBank, Step 1 corresponds to an initial setup independent of any payment. Steps 2 to 4 involve SCA authentication, up to the point where the card is loaded into the app. This process can be executed ahead of time for one or several one-time virtual cards. Steps 5 to 9 exactly correspond to a timed EMV process of payment. As a result, PrivBank results in a processing time within the range of current payment standard and similar to already deployed solutions such as tokenization or Curve. Then, from a technical point of view, PrivBank complies with the standard.

On the KYC Compliance. KYC regulations are fulfilled via the Issuer, who checks Payers' identification documents upon them opening a bank account. A contract in between the Issuer and the Proxy mandate the Issuer for the identity verification. On the other side, there is a liability-shift towards the Proxy on the verification of the Merchant's identity.

[8] *e.g.,* a Merchant M in the country, with Merchant Location and Merchant Category Code getting pseudonymised as a fixed pseudo-merchant identity

On the SCA Compliance. Strong customer authentication is checked by the Issuer when Payer ID_A intends to pay at Step 2 to provide ID_Y/C_{ID_Y} to Payer ID_A. Note that an SCA authentication (that may be carried out offline using the CDCVM tag) may be required for the payment if it is not executed within a few seconds after the first one.

On the AML Compliance. The T&Cs of `PrivBank` subscribers are such that the amount of payments that any Payer ID_A makes via `PrivBank` may be capped to values in line with the current AML legislation derogation (*e.g.,* EUR/GBP1,000). To avoid breaking the failure to disclose regulations the bank in the `PrivBank` model needs to ensure that the proxy it has partnered with is performing the required ongoing monitoring of transactions on its behalf. It would be possible to maintain the pseudonymity of the customer provided that the proxy would have access to any previous transaction data of the customer (if there is any) and be able to generate automated alerts if transactions were not in line with the customer's profile. Those alerts would be remitted to the issuer and the issuers's officers would investigate and generate a Suspicious Activity Report. In these cases of suspected fraud, the transaction would be linked to the customer's identity by the Issuer.

5.10.3 Construction `PrivProxy`

`PrivProxy` is based on the same three main parties, but when at the core of `PrivBank`'s sat an "pseudonymity-friendly" Issuer, now, in `PrivProxy`, it is a Payment Proxy which provides a service to add pseudonymity on top of EMV payments. Note that there could be many such Proxies. Figure 5.3b on page 132 summarise `PrivProxy` while Figure 5.5 on the next page describe the protocol flow.

Step 1. The Payer is known via their identity dubbed as "ID_A" in Figure 5.3b on page 132 and 5.5 on the next page by the Issuer and the Payment Proxy, and it holds accounts with both. In the onboarding process, the Payer ID_A links the bank-account that they hold with the Issuer with the user-account it holds with the Proxy.

EMV Compliance. At onboarding with `PrivProxy`, there is a banking pre-authorisation made, where the Proxy can take up to a fixed total from the Payers bank account. In line with the EMV rules, this *pre-authorisation caps* can be a maximum of x amount per year/month/day (*e.g.,* EUR/GBP1000/month). The Payer gets access to the `PrivProxy` app as a Proxy-provisioned service and has to comply to KYC procedure via their identity ID_A to use it. This time, the app, from an engineering perspective, has nothing to do with the Issuer.

Step 2. When a payment is to be made by the Payer ID_A to a Merchant M, the Payer opens the `PrivProxy` app. The opening of the app prompts the Proxy to do the SCA process, see SetupPayment in Figure 5.5 on the facing page. Upon successful SCA checks, the Proxy checks if ID_A did not reached its AML-cap. If they have, the protocol stops.

Step 3. At this stage, for Payer ID_A, the Proxy creates a one-time virtual identity and a one-time, EMV-compliant card, shown as ID_X and C_{ID_X} on Figure 5.3b on page 132 and 5.5 on the facing page. They are pseudorandom and statistically

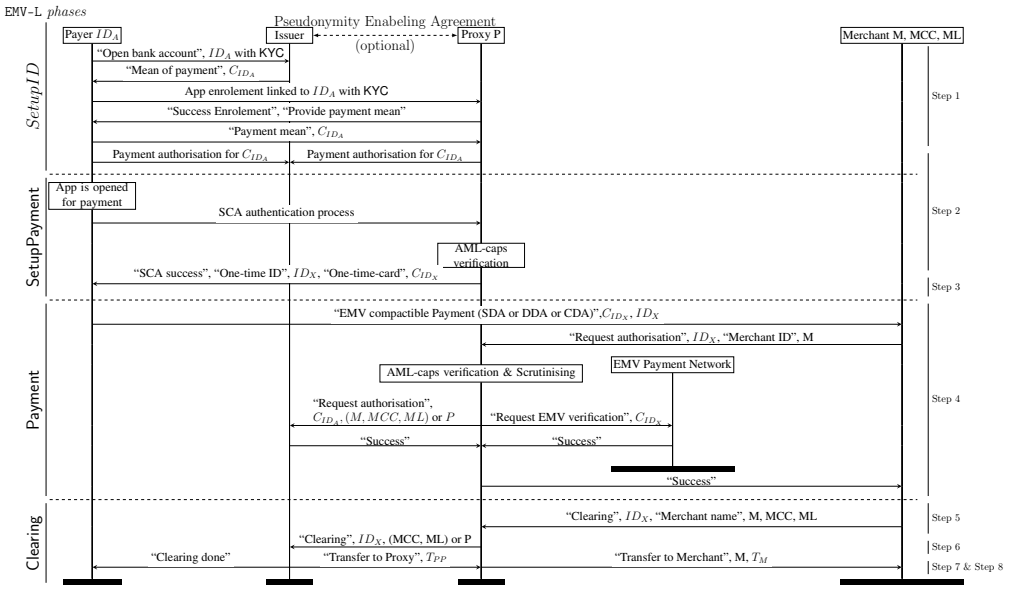


Figure 5.5: Protocol-Implementation Flow of `PrivProxy`. *All communications appart from the payment from the Payer to the Merchant are assumed to be executed on a secure channel (encrypted and authenticated communications).*

indenpendent of ID_A and C_{ID_A}. The card issued is automatically "loaded" onto the `PrivProxy` app. The transcript produced between the app and the Merchant's PoS, when paying with this app, is that of the EMV physical card except for CDCVM.

Step 4. The Payer goes to pay with it by the Merchant's PoS, as shown on Figure 5.3b on page 132 and as Payment in Figure 5.5. Accounting for the value of the current payment, the Proxy checks that the Payer's pre-authorisation caps are not reached, and nor are the AML-caps and executes the AML scrutinising.

Step 5. Using standard EMV mechanisms, the (Acquirer of the) Merchant begins to resolve the payer's payment by contacting the Proxy, which is the Issuer of C_{ID_X}.

Step 6. The Proxy behaves differently if the Merchant is on their pre-vetted, selected stores' list to use `PrivProxy` or not. In either case, the Proxy aims to provide Merchant pseudonymity and proceeds as described below.

If the Merchant is on the "selected-stores" list, then, using the payment networks' back-end, the Proxy matches ID_X with the payer's identity ID_A and goes to Issuer and asks to resolve a payment for Payer ID_A. It does not declare Merchant M but its own identity P as being the Merchant.

If the Merchant is not on the "selected-stores" list, then, using the payment networks' back-end, the Proxy does still not declare Merchant M identity to the Issuer, instead provides restricted information (*e.g.,* ML and MCC). In this case, the Proxy takes on risks in terms of their authorisation rates (*i.e.,* it is possible that the Issuer will not approve the payment due to too little information on the Merchant).

Liability and Fraud-protection. In terms of fraud-detection disputes, the Proxy takes on the liability, this is stipulated in the contract it has with the Payer, meaning it will have to reimburse the payer in some cases as it is the payment provider. Should the

Criteria	`PrivBank`	`PrivProxy`
1. Payer pseudonymity	with respect to the Proxy and the Merchant ☺	with respect to the Merchant ☹
2. Merchant pseudonymity	with respect to the Issuer ☺	with respect to the Issuer ☺
3. Liability for legal compliance	Issuer and Proxy	Proxy
4. Liability for Economic risk	Joint between Issuer and Proxy	Proxy
5. Payers' identities distributed	Yes ☹	No ☺
6. Payer's trust	In the Proxy and the Issuer	In the Proxy
7. System's assumptions	Trust between the Payment Proxy and the Issuer ☹	No ☺
8. Security assumptions	Issuer's app not leaking identities ☹	No ☺
9. Feasibility	Privacy-friendly Issuers may be rare ☹	Immediately feasible by some companies ☺
10. False Rejection Rate	None ☺	Risks of low payment authorization rates ☹

Table 5.3: `PrivBank` and `PrivProxy`: Pseudonymity-provision, Advantages and Disadvantages

Issuer need to be involved, this is done entirely by the Proxy, with no legal obligation on the Payer.

Step 7. If the payment is not authorised by the Issuer (which is unlikely for Merchants on the selected-stores list), the protocol stops. Otherwise, the Issuer checks if Payer ID_A has funds to pay and approves the transaction if it does.

Step 8. If all went through the payment is resolved towards the Proxy and then from the Proxy to the Merchant M.

5.10.4 Law Abiding and Norm Compliance Aspects of `PrivProxy`

`PrivProxy` also complies with the norm and all regulations applicable to the banking system. Arguments similar to those set out in Section 5.10.2 on page 135 apply. Some additional elements are discussed below.

On the Norm Compliance. EMV-compliant one-time card are issued and delays are managed in the same way as in `PrivBank`. Here, it is also possible to pre-load the pseudo-identity ID_X and the card C_{ID_X} in order to carry out EMV contactless payment with CDCVM authentication.

On the KYC Compliance. KYC regulations are fulfilled by the Issuer and the Proxy, who check Payers' identification documents upon them opening accounts with each.

On the SCA Compliance. Since the Proxy did KYC onboarding of the Payer, the Proxy can do the SCA step and checks that it is indeed Payer ID_A attempting to pay.

On the AML Compliance. What we called *AML-caps* may be in place, also for payments made via `PrivProxy`. In this case, it is the T&Cs of the account held with the Proxy that enforces this: the Proxy will check that the amount of payments that any Payer makes via `PrivProxy` are capped to values in line with the current AML regulations. As a payment services provider, it is the Proxy which is liable to comply with the AML and SCA frameworks. From the perspective of the bank, the relevant payment is the payment made to the Proxy. As there is no unlinkability of customer information and transaction details from the Proxy's point of view, there should be no difficulty in the Proxy generating automated alerts and Suspicious Activity Reports as required.

5.10.5 Comparing `PrivBank` and `PrivProxy`

`PrivBank` and `PrivProxy` offer different pseudonymity guarantees (see Table 5.2 on page 122). Most of the relevant comparative aspects between our two protocols are detailed in Table 5.3 on the preceding page with the relevant design choices. Specifically, entries such as rows 1 and 6 (also rows 3, 7, 8, and 10 in Table 5.3 on the facing page) illustrate that neither `PrivBank` nor `PrivProxy` can be deemed superior.

Varying Features. For instance, `PrivBank` requires additional assumptions to enhance pseudonymity, such as a legal agreement of collaboration between the Proxy and the Issuer (rows 6 and 7). Conversely, `PrivProxy` could be offered by the Proxy independently of an Issuer, though legitimate payments may be rejected (row 10) due to the lack of such an agreement and the Proxy curating Merchant-related information of their own accord. Furthermore, `PrivBank` allows for multiple Proxies as partners of the Issuers, offering Payers a choice of providers. If it is the Payers who pay for the Proxy service, then balance of trust may shift, potentially making `PrivBank` more appealing to some Payers. Also, liability is shared in `PrivBank`, but not in `PrivProxy` (row 3). `PrivBank` boasts stronger decentralisation of knowledge, however, it requires collaboration between Issuers and Proxies from the management of the app, as it necessitates a robust agreement (row 8), but –in reality– pseudonymity-friendly Issuers may be rare. Ultimately, the choice between the two proposals depends on priorities and incentives driving system deployment. Thus, we continue to present and analyse both `PrivBank` and `PrivProxy`, as they cater to different markets and operate under distinct assumptions.

Achieving $\mathcal{P}$`An`, $\mathcal{M}$`An`, `Unlnk`. `PrivBank` achieves $\mathcal{P}$`An` in front of the Proxy, which is down to Issuer and the Proxy having partial views on identifiers. This is not the case in `PrivProxy`, so $\mathcal{P}$`An` cannot be achieved there. All such results recounted in Table 1 are formalised in Section 5.11 on the next page. The information essential for pseudonymity are divided between the proxy (payment information) and the card issuer (identification information). This is necessary to comply with legislation.

On the Legal Compliance. For a summary of compliance with KYC and SCA regulations, see Table 5.2 on page 122. We require KYC from the Issuer in both `PrivBank`

and PrivProxy, but we only require it from the Payment Proxy in PrivProxy. In PrivBank, a legal agreement allows the Payment Proxy to rely on the Issuer for KYC and SCA. Thus, payments can proceed only when the Payment Proxy has received SCA approval from the Issuer. This is unlike the case of PrivProxy where the SCA is no longer required with the Issuer, as the Payment Proxy has done KYC and there is also a pre-authorisation by the Payer on their some of their funds made to the Proxy during enrolment.

On the Acceptability of the Proxy. The use of proxies in the banking system, such as Curve and Revolut, is already established and widely accepted. These services demonstrate that the concept is viable and secure, which helps mitigate concerns about trust in our solutions. Additionally, our design leverages the principle that banks do not require all merchant data for payment authorisation. This ensures that both the Issuer and the Proxy can potentially monetise the service, supporting the acceptability and feasibility of our approach.

On Implementation Aspects. Both solutions/apps generate one-time cards (*i.e.,* a single-use PAN) and run contactless EMV as for physical cards with the PoS. Loading a card mandates use of TEE to comply with the AML regulations. Alternatively, the server could load tokens associated one-time cards. Tokens do not require to be securely stored. This second scenario generates a single-use PAR, but this element would lose its traceability purposes, thus, we prefer the first solution. Apart from these minor considerations, our two solutions can be implemented on the basis of an adapted version of the services provided by existing proxies.

On the Cost Aspects. Processing payments incurs costs that are currently borne by merchants. Our proposed solutions impose greater demands on the network. The economic risks vary from one scheme to another, as shown in Table 5.3 on page 138. Consequently, this form of payment may entail higher costs, which could be absorbed by merchants, as is the norm and/or by users opting for a "premium" privacy-preserving service.

5.11 Formal Treatment of Anonymity in PrivBank and PrivProxy

Now, we will introduce a formalism that allows us to reason formally about our proposals PrivBank and PrivProxy realising the privacy notions $\mathcal{P}$An, Unlnk and $\mathcal{M}$An. This formalism is aimed to be accessible. Meanwhile, in Section 5.13 on page 149, we give a "traditional" cryptographic model and analysis of our schemes. Notably, in Section 5.13 on page 149, we show that the "simpler" definitions in this section fully capture the cryptographic definitions.

5.11.1 Execution Model

To study the security/privacy of (payment) protocols, we formalise first the parties executing them, and call them *payment parties*, or simply *parties* and denote E: *Payers*, *Issuers*, *Merchants*, and *Proxies*. These represent machines or devices or humans, associated with long-term identifiers, well-defined PPT (probabilistic polynomial-time) algorithms to execute, and they may hold cryptographic material. There can be any number of such parties. The execution of these parties gives rise to what we call *party instances*. Each party can be instantiated multiple times. This setting where party instances execute is denoted as an *execution environment*.

5.11.2 EMV-L: A Language for EMV Protocols

To describe the main parts of the algorithms onboard the payment parties, we define an API-like language (Application Programming Interface). It describes the main procedures and sub-procedures of any EMV-compliant payment protocol, hence us calling it EMV-L.

Definition 29: EMV-L: A Language for EMV Protocols

Consider a security parameter for the payment system[9]. Our *EMV-language* is formed of the following procedures:

$SetupID(ID) \to \lambda_{ID}$ sets up the part of the *execution environment* denoted here λ_{ID}, in which instances of a Payer party with the identity ID can make payments. The object λ_{ID} enables and encapsulates all payment-related algorithms as executed/related by/to the Payer identified as ID.

$\mathsf{SetupPayment}(ID) \to C$**:** based on a Payer's identity ID and its sub-environment λ_{ID}, creates an EMV-compliant *payment device* C (*e.g.,* a physical card, or a mobile-device with a card registered to it) that can transact with an EMV-compliant PoS.

$\mathsf{Payment}((ID, C), M) \to \mathsf{pay}$**:** based on an identity ID, a payment device C correctly set up as above, and a Merchant M, generates a EMV-complaint *payment transcript* pay.

$\mathsf{Clearing}(ID, M, \mathsf{pay}) \to T$**:** based on an identity ID, a Merchant M and a transaction pay produced as per the above, this procedure concludes the payment by balancing the account of the participants. It returns the *terminating data* T.

.

A classical, in-shop EMV payment or an EMV-compliant one, like our PrivBank and PrivProxy using our EMV-L language, requires the following flow of EMV-L procedures:

$$SetupID \;\to\; \mathsf{SetupPayment} \;\to\; \mathsf{Payment} \;\to\; \mathsf{Clearing}.$$

[9]The security parameter is defined by Visa, MarterCard *etc.* at the level of the system. As here, we only aim to describe the Payer's relationship with the system, we omit it from our descriptions to make the notation more accessible.

In this study, we examining the above operations from a network perspective rather than delving into cryptographic consideration. Indeed, actions like registering a payer to the Issuer does not involves cryptography. Only SetupPayment and Payment can be viewed as cryptographic protocols, respectively card's key generation and payment protocol [EMV11]. Moreover, the clearing process (Clearing) occurs between Issuers and the network lacking any unique or public specifications. Hence, we treat the above operations as black boxes.

5.11.3 Threat Model

We move on to the threat model which we consider in our subsequent analysis of PrivBank and PrivProxy.

Corruptions and Eavesdroppers. We assume that parties, Payers, Issuers, Merchants and Proxies, can be corrupted, at any point in the execution, and they can be made to behave arbitrarily. The set of corrupted parties is denoted $\mathcal{P}_{\mathsf{cor}}$. Yet, due to the auditability requirement, distinct party-types, both involved in one execution cannot be simultaneously corrupted, otherwise trivially breaking any pseudonymity property. Active Issuer and Proxy cannot be both corrupted for $\mathcal{P}\mathtt{An}$ and Unlnk, active Merchant and Proxy cannot both be corrupted for $\mathcal{M}\mathtt{An}$. Our corruptions and adversaries also vary with the properties:

For $\mathcal{P}\mathtt{An}$ and Unlnk: (1) any Payer can be corrupted except the one against whom the property is considered and at least one other; (2) an adversary can eavesdrop all payments made by the Payers to Merchants' PoS[10];

For $\mathcal{M}\mathtt{An}$: (1) any Merchant can be corrupted except the one against whom $\mathcal{M}\mathtt{An}$ is considered and at least one other. (2) An adversary can eavesdrop all payments submitted by the Proxies to the Issuers[11].

We also assume two realistic aspects. One, the (application implementing our solution on the) payer's phone will not be corrupted, unless the Payer is also corrupted. Second, other data (values, dates, *etc.*) in funds transfers are independent of the Payer's long-term identity. It means that the Payer does not encode their identity in the time or value of the payment.

5.11.4 Formalising Payments' Privacy

Our $\mathcal{P}\mathtt{An}^{C_{ID}}$ holds if, when a card C was used to make payment P, the attacker as per above cannot build a (mathematical) relation of the type "payment P is related to a card C". Below, we formally define relations to describe $\mathcal{P}\mathtt{An}$, Unlnk, and $\mathcal{M}\mathtt{An}$, starting with some notions on mathematical relations.

[10]This particularly pertinent in the case where the channel between the PoS and Payer is public/unencrypted, which is the case today. In this setting, the eavesdropper attacker is weaker than a corrupted Merchants/Payers. However, there are proposals to make this channel secure in EMV 2nd Gen [EMV14].

[11]This is a very strong attacker in practice, as it would mean that there is breach in the backend of the payment networks.

Preimage Resistant. Let T be a function which image is of size a security parameter. We say it is *preimage resistant* if for all $v \in Im(T)$ (the image of the function T), finding $T^{-1}(v)$ is hard to compute it with non-negligible probability for all PPT algorithm.

Class Hiding. Let T be a binary relation on pairs (x, x'). This relation is *class hiding* if for two elements x and x', it is hard to determine if (x, x') belongs to T.

Let $\cdot^{\mathcal{E}} \colon \mathsf{PAY} \to \mathsf{PAY}^{\mathcal{E}}$ be a *restriction function* associating a view of a payment transcript pay in the set of all possible payment transcripts PAY to a restricted transcript $\mathsf{pay}^{\mathcal{E}}$ for the parties in $\mathcal{E} \subset E$. To formalise $\mathcal{P}\texttt{An}$, we introduce a *Payer relation*, linking the Payer's long-term identity ID to payment transcripts pay generated in any EMV protocol (described in EMV-L) and its restricted version.

Definition 30: (Restricted) Payer Relation

Let π be an EMV protocol described in EMV-L. Let ID be a set of at least two long-term Payers' identifiers. Let PAY be a list of outputs of the protocol $\mathsf{Payment}$ in π, generated by the Payers with identifiers in ID. We define the *Payer relation* $\mathcal{R}_{\mathcal{P}\mathsf{Idt}} \subsetneq \mathsf{ID} \times \mathsf{PAY}$ as $(ID, \mathsf{pay}) \in \mathcal{R}_{\mathcal{P}\mathsf{Idt}}$ if

$$\exists \lambda \in [SetupID(ID)], \exists C \in [\mathsf{SetupPayment}(ID)], \mathsf{pay} \in [\mathsf{Payment}((ID, C), M)].$$

Given a subset $\mathcal{E} \subset E$, we can now restrict the relation to the view of parts of the entities that have been involved in the EMV protocol. We define the *restricted Payer relation* $\mathcal{R}^{\mathcal{E}}_{\mathcal{P}\mathsf{Idt}}$ as

$$(ID, \mathsf{pay}^{\mathcal{E}}) \in \mathcal{R}^{\mathcal{E}}_{\mathcal{P}\mathsf{Idt}} \text{ if } (ID, \mathsf{pay}) \in \mathcal{R}_{\mathcal{P}\mathsf{Idt}}.$$

Assuming that no two transactions within any of the protocol's views leads to identical data transcript (due to the same timestamp amongst others), then $\cdot^{\mathcal{E}}$ is a bijection and the elements in $\mathcal{R}^{\mathcal{E}}_{\mathcal{P}\mathsf{Idt}}$ are in bijection with the elements in $\mathcal{R}_{\mathcal{P}\mathsf{Idt}}$.

We now define the *payments relation*, which enable us to define $\texttt{Unlnk}$: it denotes a relationship associating two payment transcripts, pay and pay', when they are executed by the same Payer, in any EMV protocol (described in EMV-L).

Definition 31: (Restricted) Payments Relation

Let π be an EMV protocol described in EMV-L. Let ID be a set of, at least two, long-term identifiers. Let PAY be a list of outputs of the protocol $\mathsf{Payment}$ executed in π by Payers with identifiers in ID. We define the *payments relation* $\mathcal{R}_{\texttt{Payms}} \subsetneq \mathsf{PAY} \times \mathsf{PAY}$ as follows: $(\mathsf{pay}, \mathsf{pay}') \in \mathcal{R}_{\texttt{Payms}}$ if

$$\exists ID, M, M', \exists \lambda \in [SetupID(ID)], \exists C, C' \in [\mathsf{SetupPayment}(ID)],$$
$$\mathsf{pay} \in [\mathsf{Payment}((ID, C), M)], \text{ and } \mathsf{pay}' \in [\mathsf{Payment}((ID, C'), M')].$$

Given a subset $\mathcal{E} \subset E$, the *restricted payments relation* $\mathcal{R}^{\mathcal{E}}_{\texttt{Payms}}$ is defined by

$$(\mathsf{pay}^{\mathcal{E}}, \mathsf{pay}'^{\mathcal{E}}) \in \mathcal{R}^{\mathcal{E}}_{\texttt{Payms}} \text{ if } (\mathsf{pay}, \mathsf{pay}') \in \mathcal{R}_{\texttt{Payms}}.$$

Similarly to the *payer relation*, which refers to Payers, we introduce a *Merchant relation*.

Definition 32: (Restricted) Merchant Relation

Let π be an EMV protocol described in EMV-L. Let $\mathcal{M}$ be a set of at least two Merchant identifiers and PAY a set of payments for some $M \in \mathcal{M}$. We define the *Merchant relation* $\mathcal{R}_{\mathcal{M}\mathsf{Idt}} \subsetneq \mathcal{M} \times \mathsf{PAY}$ as

$$(M, \mathsf{pay}) \in \mathcal{R}_{\mathcal{M}\mathsf{Idt}} \text{ if } \exists ID, \exists \lambda \in [SetupID(ID)], \exists C \in [\mathsf{SetupPayment}(ID)], \\ \mathsf{pay} \in [\mathsf{Payment}((ID, C), M)].$$

We define the *restricted Merchant relation* for a set $\mathcal{E}$ of parties as

$$(M, \mathsf{pay}^{\mathcal{E}}) \in \mathcal{R}^{\mathcal{E}}_{\mathcal{M}\mathsf{Idt}} \text{ if } (M, \mathsf{pay}) \in \mathcal{R}_{\mathcal{M}\mathsf{Idt}}.$$

Next, by requiring intractability of properties on these relations, define our three payments-privacy properties.

Payer Pseudonymity. The *pseudonymity with respect to the identity of the Payer,* $\mathcal{P}\boldsymbol{An}^{ID}$, requires that Payer's long-term identity remains unknown during the payment from the set of parties $\mathcal{E}$. We model this using the preimage resistance of the identification relation $\mathcal{R}^{\mathcal{E}}_{\mathcal{P}\mathsf{Idt}}$ (see Definition 33).

Definition 33: Pseudonymity with respect to the identity - $\mathcal{P}\texttt{An}^{ID}$

Let $\mathcal{R}_{\mathcal{P}\mathsf{Idt}}$ be an identification relation defined by an EMV payment protocol described in EMV-L and $\mathcal{E}$ be a set of parties. We say the *protocol attains* $\mathcal{P}\boldsymbol{An}^{ID}$ *in front of a set* $\mathcal{E}$ *of parties* if the relation $\mathcal{R}^{\mathcal{E}}_{\mathcal{P}\mathsf{Idt}}$ is preimage resistant: *i.e.,* it is intractable to yield ID as ${\mathcal{R}^{\mathcal{E}}_{\mathcal{P}\mathsf{Idt}}}^{-1}(\mathsf{pay}^{\mathcal{E}})$.

In Section 5.5.2 on page 120, we introduced pseudonymity with respect to Payers' long-term card data C_{ID}, and with respect to Payers' identity ID. In the EMV-L language, the object C which is the output of $\mathsf{SetupPayment}(ID)$ is in fact a tuple $C = (C_{ID}, C_{short})$, where C_{ID} corresponds to long-term card data and C_{short} to ephemeral/one-time card data.

To introduce $\mathcal{P}\texttt{An}^{C_{ID}}$, we define a relation $\mathcal{R}_{\mathcal{C}Idt}$ replacing the identity ID in the Payer relation by the card C_{ID}, then, replacing in Definition 33 the preimage resistance of $\mathcal{R}_{\mathcal{P}\mathsf{Idt}}$ with the preimage resistance of $\mathcal{R}_{\mathcal{C}\mathsf{Idt}}$ gives us what *pseudonymity* $\mathcal{P}\boldsymbol{An}^{C_{ID}}$ *with respect to Payers' long-term card data* C_{ID} is.

Unlinkability. *Unlinkability* in payments refers to the capacity to ascertain whether two payments originate from the same Payer. It is based on the class hiding property of $\mathcal{R}^{\mathcal{E}}_{\texttt{Payms}}$.

Definition 34: Unlinkability - `Unlnk`

Let $\mathcal{R}_{\texttt{Payms}}$ be a payments relation defined by a EMV protocol and $\mathcal{E}$ be a set of parties. We say that the protocol *attains unlinkability* $\boldsymbol{\texttt{Unlnk}}$ *for a set* $\mathcal{E}$ *of parties* if $\mathcal{R}^{\mathcal{E}}_{\texttt{Payms}}$ is class hiding.

Merchant Pseudonymity. Merchant pseudonymity is defined similarly to the Payer's pseudonymity $\mathcal{P}\texttt{An}$. It is also based on the preimage resistance, but, this time, of a *Merchant relation* $\mathcal{R}_{\mathcal{M}\mathsf{Idt}}$.

Definition 35: Merchant Pseudonymity - $\mathcal{M}\texttt{An}$

Let $\mathcal{R}_{\mathcal{M}\mathsf{Idt}}$ be a Merchant relation defined by a payment protocol and $\mathcal{E}$ be a set of parties. We say that the protocol *attains Merchant pseudonymity* $\boldsymbol{\mathcal{M}\texttt{An}}$ *for a set* $\mathcal{E}$ *of parties* if the relation $\mathcal{R}^{\mathcal{E}}_{\mathcal{M}\mathsf{Idt}}$ is preimage resistant.

We note that $\mathcal{M}\texttt{An}$ can also recast into *partial Merchants' pseudonymity* $\mathcal{M}^{part}_{\texttt{An}}$, whereby partial preimaging of $\mathcal{R}^{\mathcal{E}}_{\mathcal{M}\mathsf{Idt}}$ is possible. That leads to saying that a Merchant is with subset of the domain of $\mathcal{R}_{\mathcal{M}\mathsf{Idt}}$, *i.e.,* it is a Merchant with a MCC and ML shared by several.

5.11.5 Provable Anonymity in `PrivBank` and `PrivProxy`

Next, we state and prove our properties $\mathcal{P}\texttt{An}$, $\mathcal{M}\texttt{An}$, and `Unlnk` for `PrivBank` and `PrivProxy`.

As per our threat model, all observer between the Payer and the Merchant is at most as strong as a corrupt Merchant. Similarly, someone who breaks the back-end channel between the Merchant and the Issuer is at most as strong as a corrupt Issuer. So, we state our results below only with respect to corrupt parties.

We start with the pseudonymity of the Payer, which differs in our constructions. Indeed, a KYC procedure is required in `PrivProxy`, where the pseudonymity-friendly Issuer provides a one-time identity for the Payer to present to the Payment Proxy in `PrivBank`. This KYC-based difference indirectly leads to:

Property 8: `PrivBank` – $\mathcal{P}\texttt{An}^{ID}$ and $\mathcal{P}\texttt{An}^{C_{ID}}$

Consider an arbitrarily picked honest Payer with identifier ID, and `PrivBank` in the threat model given, where Issuers which gives service to Payer ID is being honest. Then, `PrivBank` attains $\mathcal{P}\texttt{An}^{ID}$ and $\mathcal{P}\texttt{An}^{C_{ID}}$ in front of the Proxy and the Merchant.

Property 9: `PrivProxy` – $\mathcal{P}\texttt{An}^{ID}$ and $\mathcal{P}\texttt{An}^{C_{ID}}$

Consider an arbitrarily picked honest Payer with identifier ID, and `PrivProxy` in the threat model given, where the Issuers and Proxies which give joint service to Payer ID are being honest. Then, `PrivProxy` attains $\mathcal{P}\texttt{An}^{ID}$ and $\mathcal{P}\texttt{An}^{C_{ID}}$ in front of the Merchant.

We move to the attainment of payments' unlinkability. In PrivProxy, the Proxy can link payments by the virtue of the fact that it is controlling all sides of identifiers of the payers. In PrivBank, even if the Proxy does not know the Payer's ID_A, the Proxy can link their payments by using the Android/Apple account in the phone of the Payer for which is receives requests. We change this with respect to the app (especially since it is provisioned by the Issuer). However this would complicate the resolution/authorisation of payments by the Proxy towards the Issuer. This leads us to the result below.

Property 10: PrivBank and PrivProxy– Unlnk

Consider an arbitrarily picked honest Payer with identifier ID, and PrivBank and PrivProxy in the threat model given, where the Issuers and Proxies which give joint service to Payer ID are not corrupt. Then, PrivBank and PrivProxy attain Unlnk in front of the Merchant.

We move to merchants' pseudonymity. Our result is:

Property 11: PrivBank and PrivProxy– $\mathcal{M}$An

Consider an arbitrarily picked honest Merchant with identifier M, and PrivBank and PrivProxy in the threat model given, where the Proxies and Payers which jointly pays to Merchant M are being honest. Then, PrivBank and PrivProxy attain $\mathcal{M}$An in front of the Issuer.

For the benefit of the Payer, the Merchant's pseudonymity is guaranteed in both our schemes against corrupt Issuers. AML regulations still require the identity of the Merchant to be known to at least one party for full auditability purposes. As the Payment Proxy provides the means of payment, it is contacted by the Merchant via the EMV-compliant payment validation process and knows its identity. Thereforce, the Payment Proxy knows the link between the payment and the Merchant.

All proofs are provided in Section 5.12.

5.12 Proofs for Our Main Results in Section 5.11.5

In this section, we give proofs of the privacy preserving properties, given in Section 5.11.5 on the preceding page, and achieved by our two constructions PrivBank and PrivProxy. These properties are also summarised in Table 5.2 on page 122.

Our properties consist of straightforward characteristics that can be elucidated through a high-level description, as provided in Section 5.10 on page 131. In our designs, we avoid delving into the cryptographic intricacies within the protocols. Instead, we treat them as black boxes, particularly in the payment and clearing processes. The payment mechanisms in our proposals remain unaltered compared to the extensively analyzed EMV payment procedures between the card-issuing entity (which is the Proxy in our cases), a merchant's PoS and a Payer with its card (see Section 5.3 on page 117 for related protocol analysis). Furthermore, the backend infrastructure lies outside the public specification of EMV standards and varies among different entities. As the emphasis is on the non-disclosure of certain identity-related information, we trated them as black box

only formulating a few direct hypotheses in our Threat Model (see Section 5.11.3 on page 142).

While our proofs may appear somewhat high-level to formal-methods experts, they demonstrate our pseudonymity properties of our protocols within the current EMV norms, regulations and under realistic security assumptions for EMV. Indeed, they primarily consist of simple arguments under already taken EMV security assumptions. These arguments can be thought of as *"a piece of information x is not available to the adversary, and it will not be able to recover it from other parts of the transcript, hence it will not be able to learn x."* For a more in-depth discussion and the parallel with a game-based formalism, please refer to Section 5.13 on page 149.

Given by the EMV-L description of `PrivBank` and `PrivProxy`, we need to show: for Property 1 and Property 2 – that $\mathcal{R}_{\mathcal{P}\mathtt{Idt}}$ and $\mathcal{R}_{\mathcal{C}\mathtt{Idt}}$ are preimage resistant; for Property 3 – to show that $\mathcal{R}_{\mathtt{Payms}}$ is class hiding; Property 4 – that $\mathcal{R}_{\mathcal{M}\mathtt{Idt}}$ is preimage resistant.

Proof of Property 8, `PrivBank` $\mathcal{P}$`An` . We show the payer pseudonimity $\mathcal{P}\mathtt{An}$ for `PrivBank` as stated in Property 8 on page 145. We need to show that $\mathcal{R}_{\mathcal{P}\mathtt{Idt}}$ and $\mathcal{R}_{\mathcal{C}\mathtt{Idt}}$ given by the EMV-L description of `PrivBank` are preimage resistant.

For our analysis, we refer to Figure 5.4 on page 133, which provides a graphical description of `PrivBank` identity information exchanges.

Payer/Card Pseudonymity with respect to the Proxy. Messages sent to the Payment Proxy contain an application registration and the identity ID_X obtained from the Issuer. By assumption in the proposition, the Issuers which interact with Payer ID are not corrupt. So, the ID_X is generated indeed as a unique identity independent of ID_A and hiding ID_A. Also, our threat model guarantees that the app is uncorrupted, so data on it, including ID_X and ID_A, is correctly separated between parties.

So, the payment Payment Proxy has no advantage in reversing the $\mathcal{R}_{\mathsf{PAY}}$ payment relationship even if it has observed the payments executed based on credentials it has provided. The same is true of C_{ID_X} and C_{ID_A}.

Payer/Card Pseudonymity with respect to the Merchant. The Issuer is not corrupted. So, the Issuer does not give ID_A (resp. C_{ID_A} for card pseudonymity). As before, by assumption in the proposition, the Issuers which interact with Payer ID are not corrupt. So, the ID_X is also generated indeed as a unique identity independent of ID_A and hiding ID_A. So, all is left is for the Merchant to guess ID_A (resp. C_{ID_A} for card pseudonymity) on the basis of a payment derived solely from identity ID_Y and card C_{ID_Y} both provided by the Payment Proxy. Consequently, the Merchant is unable to retrieve the Payer's identities if the Payment Proxy is unable to do so.

Both corrupted entities receive only transaction information T_{PP} from the Issuer which are unlinked to the Payer's identity ID_A, when executed under identity ID_X (by construction). ID_A is also not encoded in details in the payments' date or value (by the threat model on the payer).

The impossibility of reversing the $\mathcal{R}^{\mathcal{P}_{\mathsf{cor}}}_{\mathsf{PAY}}$ payment relationship and then $\mathcal{P}\mathtt{An}^{ID}$ and $\mathcal{P}\mathtt{An}^{C_{ID}}$ pseudonymity stems from all the above. □

Proof of Property 9, PrivProxy $\mathcal{P}$An . We show the payer pseudonimity $\mathcal{P}\texttt{An}$ for our second protocol PrivProxy as stated in Property 9 on page 145.

We need to show that $\mathcal{R}_{\mathcal{P}\texttt{Idt}}$ and $\mathcal{R}_{\mathcal{C}\texttt{Idt}}$ given by the EMV-L description of PrivProxy are preimage resistant.

For our analysis, we refer to Figure 5.5 on page 137, which provides a graphical description of PrivProxy identity information exchanges.

Payer/Card Pseudonymity with respect to the Merchant. The Merchant first receives a payment based on ID_X and C_{ID_X}. In this construction, both ID_A and C_{ID_A} are sent to the Payment Proxy, the entity by the Issuer. But, by as per assumption in the proposition, the Issuers and Proxies which give joint service to Payer ID are not corrupt, and the app is not corrupt, so then ID_X and C_{ID_X} are generated by the Proxy and remain independent of and not linked to the Payer long-term information ID_A and C_{ID_A}.

The Merchant also receives T_M as the final payment for the purchase, which is also not linked to long-term identity or card data (by the threat model on the Payer not encoding his ID in the values/dates of the payments).

So, given all this independence of data from ID_A and C_{ID_A}, the impossibility of reversing the payment relationship $\mathcal{R}_{\mathsf{PAY}}^{\mathcal{P}_{\mathsf{cor}}}$ and then $\mathcal{P}\texttt{An}^{ID}$ and $\mathcal{P}\texttt{An}^{C_{ID}}$ pseudonymity follows.

Thus, pseudonymity properties $\mathcal{P}\texttt{An}^{ID}$ and $\mathcal{P}\texttt{An}^{C_{ID}}$ are achieved for PrivProxy. □

Proof of Property 10, PrivBank and PrivProxy Unlnk . We treat both proposal in the same time and show the payer unlinkability Unlnk for PrivBank and PrivProxy as stated in Property 10 on page 146.

We need to prove that $\mathcal{R}_{\texttt{Payms}}$ given by the EMV-L description of our proposals is class hiding.

We treat both PrivBank and PrivProxy at once.

Unlinkability with respect to Merchant. First, the Merchant gets a payment based on a single use identity and card ID_X/C_{ID_X} or, respectively, ID_Y/C_{ID_Y} independent of the original identity ID_A or on a long-term card C_{ID_A}. This independence is ensured by the way we have chosen to generate the one-time identities and by the fact that the threat model guarantees that the app is uncorrupted, so data on it is correctly separated between parties

The second interaction which goes to the Merchant corresponds to the fund transfer carried out by the Payment Proxy and is also independent of the Payer's identity (by assumption).

So, $\mathcal{R}_{\texttt{Payms}}$ given by the EMV-L description of our proposals is class hiding with respect to the Merchant and Unlnk unlinkability is therefore achieved. □

Proof of Property 11, PrivBank and PrivProxy $\mathcal{M}$An . We treat both proposal in the same time and show the merchant anonymity $\mathcal{M}\texttt{An}$ for PrivBank and PrivProxy as stated in Property 11 on page 146.

First, we notice that no direct contact occurs between the Merchant and the Issuer. Also, the only message coming from the Merchant and later going to the Issuer appends with the payment validation (only in `PrivBank`) and the clearing (in both protocols).

Considering that the Payment Proxy sanitizes the Merchant identity by removing any Merchant dependent information and replace it with its identity, as we took as a hypothesis that Proxies which jointly pay the Merchant M are not corrupt. It should be noted that our protocols provide for an alternative procedure where the Payment Proxy provides partial information such as MCC and ML to the Issuer. This alternative procedure will achieve $\mathcal{M}\texttt{An}$ with respect to Merchants who share the same MCC and ML data. We now conclude our proof, which continues in the same way in both cases, but which results in different pseudonymity properties.

From the honest Payment Proxy's sanitization, we have that no Merchant's identifying information (or partial but not identifying) is passed on from the Payment Proxy to the Issuer. Hence, the adversary is left trying to find a preimage for the relation $\mathcal{R}^{\mathcal{E}}_{\texttt{Payms}}$, only based on data unrelated to the merchant identity. This allows us to conclude that $\mathcal{R}^{\mathcal{E}}_{\texttt{Payms}}$ is preimage resistance, hence, `PrivBank` and `PrivProxy` achieves $\mathcal{M}\texttt{An}$ in the current corruption setting. □

5.13 Game Based Formalisation

Below, we recast our security definitions in the widely-accepted game-based paradigm. These definitions demonstrate the security of our proposal in the cryptographic game based standard. Multiple formalisations for game-based properties, with varying degrees of strength are possible. The game based definitions which the first definition is for Payer's pseudonymity and unlinkability and the second for Merchant's pseudonymity are directly implied by the security introduced through relations in Section 5.11 on page 140. De facto showing that our protocols matches the security defined in this section.

We have chosen to present only one game for the Payer's privacy preserving properties, as in general unlinkability `Unlnk` imply pseudonymity $\mathcal{P}\texttt{An}$ (*i.e.,* $\texttt{Unlnk} \implies \mathcal{P}\texttt{An}$). In all the upcoming games, an EMV compatible payment procedure is formalised through our EMV language (see Definition 29 on page 141) and executed by a challenger following the games against an adversary $\mathcal{A}$. This adversary is assumed to corrupt and execute entities participating in the protocols in the experiments following the threat model of Section 5.11.3 on page 142. All corrupted entities are encompassed in a set called $\mathcal{P}_{\mathsf{cor}}$.

Payer's Payments Privacy: Pseudonymity + Unlinkability *Payer Pseudonymity* models that entities does not retrieve the Payer's long-term card data (card number, expiry date, CVV, *etc.*) nor its identity during the execution of payments. *Payments' unlinkability* models a stronger requirement against an adversary trying to match payments by distinguishing between cases where they were made by the same entity or not.

Below, we present an experiment encompassing both notions of pseudonymity and unlinkability. The adversary has access to a payment oracle and a clearing oracle which

allows him on one side to link the payment under consideration or to infer the identity of the Payer based on the other executions it sees. Another mean of attack would be to only base itself on the payment in consideration. The latter relates to pseudonymity, while the first refer to the unlinkability property. Before introducing the experiment, let us set up the oracles in full details.

Payment. The oracle $\mathsf{Payment}((\cdot, C.), \cdot)$, is queried by an adversary to the challenger on the basis of a Payer identity ID and a Merchant identity M. The identity of the Payer must have been initialised prior to this call, meaning that: the $SetupID(ID)$ and $\mathsf{SetupPayment}(ID)$ protocols, which returned a C_{ID} card, have been executed. It executes the protocol $\mathsf{Payment}((ID, C_{ID}), M)$ with the adversary in the corruption frame under consideration.

Clearing. The oracle $\mathsf{Clearing}(\cdot)$, is queried by an adversary for a payment pay. The payment pay must have been previously produced on the basis of the $\mathsf{Payment}$ oracle. It executes the $\mathsf{Clearing}(\mathsf{pay})$ protocol with the adversary in the corruption frame under consideration.

Definition 36: Game Based Unlinkability with Pseudonymity

Consider a security parameter for the payment protocol. Consider the set of oracles $\mathcal{O} = (\mathsf{Payment}((\cdot, C.), M), \mathsf{Clearing}(\cdot, M, \mathsf{pay}))$, each of them operating as defined above. Consider a PPT adversary $\mathcal{A}$ participating in all executed protocols under the role of the corrupted entities $\mathcal{P}_{\mathsf{cor}}$ against a challenger executing the actions prescribed by $\mathsf{Exp}^{\mathrm{PRIV}}$ for uncorrupted entities.

$\mathsf{Exp}^{\mathrm{PRIV}}_{\mathcal{P}_{\mathsf{cor}}}$

1 : $\{ID_i\}_{i=1}^n, M \leftarrow \mathcal{A}$
2 : **Let** $T = \emptyset$ the set of initialised cards.
3 : **for** $i \in \{1, \ldots, n\}$,
4 : $\quad \lambda_i \leftarrow SetupID(ID_i)$
5 : $\quad C_i \leftarrow \mathsf{SetupPayment}(ID_i)$
6 : $\quad T \leftarrow T \cup \{ID_i, C_i\}$
7 : Sample an identity with the associated card $(ID, C) \xleftarrow{\$} T$
8 : $\mathsf{pay} \leftarrow \mathsf{Payment}((ID, C), M)$
9 : $f \leftarrow \mathsf{Clearing}(\mathsf{pay})$
10 : **return** $ID^* \leftarrow \mathcal{A}^{\mathcal{O}}$

We say that a payment scheme has *game based unlinkability with pseudonymity for a set of corrupted entities* $\mathcal{P}_{\mathsf{cor}}$ if for adversaries $\mathcal{A}$ controlling entities in $\mathcal{P}_{\mathsf{cor}}$,

$$|\Pr[\mathsf{Exp}^{\mathrm{PRIV}}_{\mathcal{P}_{\mathsf{cor}}}(\mathcal{A}) = ID] - \frac{1}{n}| \leq \epsilon,$$

for an negligible ϵ in the security parameter of the EMV system.

An alternative definition would require $\mathsf{Exp}^{\mathrm{PRIV}}$ to return a long term card. This definition is analogue but apply for *Pseudonymity with respect to long-term card* $\mathcal{P}\mathtt{An}^{C_{ID}} \subset \mathsf{CARD} \times \mathsf{PAY}$ and requires the opponent to return the Payer's long-term card C_{ID} instead of the Payer's identity ID. We call this experiment $\mathsf{Exp}^{\text{PRIV-}C_{ID}}$, it is associated with the same winning probability.

Theorem 3

Assuming that the relation $\mathcal{R}^{\mathcal{P}_{\mathsf{cor}}}_{\mathsf{PAY}} \subsetneq \mathsf{ID} \times \mathsf{PAY}$ is preimage resistant and that the relation $\mathcal{R}^{\mathcal{P}_{\mathsf{cor}}}_{\mathtt{Unlnk}} \subsetneq \mathsf{PAY} \times \mathsf{PAY}$ is class hiding for a payment protocol described by an EMV language and for a set of corrupted entities, $\mathcal{P}_{\mathsf{cor}}$ then game base unlinkability is achieved for $\mathcal{P}_{\mathsf{cor}}$.

Proof. Let $\mathcal{A}$ be an adversary executing the corrupted entities from $\mathcal{P}_{\mathsf{cor}}$ against the challenger of the experiment $\mathsf{Exp}^{\mathrm{PRIV}}_{\mathcal{P}_{\mathsf{cor}}}$.

$\mathcal{A}$ has access to oracles $\mathsf{Payment}((ID, C), M)$ and $\mathsf{Clearing}(ID, M, \mathsf{pay})$ for all registered entities and for the identities it has chosen.

The execution scenario defined by the adversary's calls to the oracles and the prescribed execution of $SetupID$, SetupPayment, Payment and Clearing define a Payment Relation $\mathcal{R}_{\mathsf{PAY}}$ and a linkability relation $\mathcal{R}_{\mathtt{Unlnk}}$. For both relation, the adversary's against these relations are its restricted counterpart: $\mathcal{R}^{\mathcal{P}_{\mathsf{cor}}}_{\mathsf{PAY}}$ and $\mathcal{R}^{\mathcal{P}_{\mathsf{cor}}}_{\mathtt{Unlnk}}$.

In the experiment, the adversary has two ways of guessing the Payer's identity. It can either try to retrieve the identity ID from its view of the payment transcript associated to pay, or try to deduce it on the basis of any links it might find with the other executions called through the oracles and then infer the identity based on the knowledge of the inputted identities. As $\mathcal{A}$ chooses the identity ID for its calls to the oracles, the relation between the transcript it sees and the Payer's identities is not hidden from it.

The first case refers to an adversary recovering the identity from the payment pay:

Consider an adversary $\mathcal{A}$ executing entities in $\mathcal{P}_{\mathsf{cor}}$ and interacting with the challenger of $\mathsf{Exp}^{\mathrm{PRIV}}$. $\mathcal{A}$ has access to oracles $\mathsf{Payment}((ID, C), M)$ and $\mathsf{Clearing}(ID, M, \mathsf{pay})$ in addition to the view it gets from the executions of $SetupID$, SetupPayment, Payment and Clearing. Based on the above algorithms, two sets can be populated: ID and PAY. The link between the elements of these sets, obtained based on the challenger's view of the execution allows defining the associated payment relation $\mathcal{R}_{\mathsf{PAY}}$. Based on the corruption set $\mathcal{P}_{\mathsf{cor}}$ and the relation, the restricted relation $\mathcal{R}^{\mathcal{P}_{\mathsf{cor}}}_{\mathsf{PAY}} \subsetneq \mathsf{ID} \times \mathsf{PAY}^{\mathcal{P}_{\mathsf{cor}}}$ is defined for the adversary's execution scenario.

Assume $\mathcal{A}$ returns a response ID^* for its view $\mathcal{R}^{\mathcal{P}_{\mathsf{cor}}}_{\mathsf{PAY}}$ of the relation $\mathcal{R}_{\mathsf{PAY}}$ and has probability significantly different from $1/n$ to win the experiment. When the adversary wins the experiment and returns the right answer $\mathcal{A}(\mathsf{Exp}^{\mathrm{PRIV}}) \to ID^*$ for the execution scenario, this means that $(ID^*, \mathsf{pay}) \in \mathcal{R}_{\mathsf{PAY}}$. As this is assumed to be with non-negligible probability, hence we invert $\mathcal{R}^{\mathcal{P}_{\mathsf{cor}}}_{\mathsf{PAY}}$ with non-negligible probability.

If the adversary provides a correct ID^*, for the payment pay, we can build a simulator algorithm based on the adversary to break the preimage resistance of $\mathcal{R}_{\mathsf{PAY}}$. This lead to a contradiction under the assumption that the relation $\mathcal{R}^{\mathcal{P}_{\mathsf{cor}}}_{\mathsf{PAY}}$ is difficult to inverse.

We may now consider the second case: assume that $\mathcal{A}$ made a link between pay and one of the payment it has sees during it call to the oracles. As the adversary knows the link between the second payment and the Payer making it, this lead to a link between pay and the identity ID^*. If the adversary provides a correct ID^*, which is the input of a Payment oracle query or a *Clearing* oracle query, we can build a simulator based on $\mathcal{A}$ breaking the class hiding property $\mathtt{Unlnk}$. This also leads to a contradiction, as we have assumed $\mathcal{R}^{\mathcal{P}_{\mathsf{cor}}}_{\mathtt{Unlnk}}$ to be class hiding resistant.

Both potential attacks are covered, hence, under preimage resistance of $\mathcal{R}^{\mathcal{P}_{\mathsf{cor}}}_{\mathsf{PAY}} \subsetneq \mathsf{ID}\times\mathsf{PAY}$ and class hiding resistance of $\mathcal{R}^{\mathcal{P}_{\mathsf{cor}}}_{\mathtt{Unlnk}} \subsetneq \mathsf{PAY}\times\mathsf{PAY}$ then game based unlinkability is achieved. □

Remarks on the Proof. One can view this proof as a game hop. First, a random permutation is introduced over the identities that $\mathcal{A}$ queries to the oracle, while still expecting the original identity as a response. This constitutes the first step of the proof, where we reduce security to pseudonymity $\mathcal{P}\mathtt{An}$. The second part of the proof involves a direct reduction from the modified experiment to unlinkability $\mathtt{Unlnk}$.

Long-term card data $\mathcal{P}\mathtt{An}^{C_{ID}}$. A similar theorem and proof apply for *Pseudonymity with respect to long-term card data* associated to the experiment $\mathsf{Exp}^{\text{PRIV-}C_{ID}}$ by requiring the opponent to return a long-term card instead of an identity. In this latter case, the same implication holds and the security based on the basic game model is also ensured for our two protocols under the associated corruption sets $\mathcal{P}_{\mathsf{cor}}$.

Merchant Pseudonymity ($\mathcal{M}\mathtt{An}$). Merchant pseudonymity guarantees the Payers that the identity of the Merchant they are paying is not disclosed to other entities during the payment or its clearing. First, let us set up the oracles useful for our definition.

SetupID. The oracle $SetupID(\cdot)$ is queried by an adversary from the challenger on the basis of the identity of a Payer ID. The challenger executes the protocol $SetupID(ID)$ interacting with the adversary in the considered corruption frame.

SetupPayment. The oracle $\mathsf{SetupPayment}(\cdot)$, is queried by an adversary from the challenger on the basis of an identity Payer's identity ID which should have been initialised based on $SetupID$. The challenger executes the protocol, $SetupID(ID)$ interacting with the adversary in the considered corruption frame.

Payment & Clearing. These oracles are the same as before.

Definition 37: Game Based Merchant Pseudonymity

Let $\mathcal{P}_{\mathsf{cor}}$ be a set of identities describing the corruption setup and consider the set of oracles $O_{\mathcal{M}\mathtt{An}} = (SetupID(\cdot), \mathsf{SetupPayment}(\cdot), \mathsf{Payment}((\cdot, C), M), \mathsf{Clearing}(\cdot, M, \cdot))$, each of them operating as defined above. Consider a PPT adversary $\mathcal{A}$ participating in all executed protocols under the role of the corrupted entities $\mathcal{P}_{\mathsf{cor}}$ against a challenger executing the actions prescribed by $\mathsf{Exp}^{\mathcal{M}\mathtt{An}}$ for uncorrupted entities.

$\mathsf{Exp}^{\mathcal{M}\mathtt{An}}_{\mathcal{P}_{\mathsf{cor}}}$

1 : $ID, M_0, M_1 \leftarrow \mathcal{A}$
2 : $\lambda \leftarrow SetupID(ID)$
3 : $C \leftarrow \mathsf{SetupPayment}(ID)$
4 : $b \xleftarrow{\$} \{0,1\}$
5 : $\mathsf{pay} \leftarrow \mathsf{Payment}((ID, C), M_b)$
6 : $f \leftarrow \mathsf{Clearing}(ID, M_b, \mathsf{pay})$
7 : **return** $M^* \leftarrow \mathcal{A}^{\mathcal{O}_{\mathcal{M}\mathtt{An}}}$

A payment scheme has *game Merchant pseudonymity for a corruption set* $\mathcal{P}_{\mathsf{cor}}$ if for any adversaries $\mathcal{A}$ executing entities in $\mathcal{P}_{\mathsf{cor}}$,

$$|\Pr[\mathsf{Exp}^{\mathcal{M}\mathtt{An}}_{\mathcal{P}_{\mathsf{cor}}}(\mathcal{A}) = M_b] - \frac{1}{2}| \leq \epsilon,$$

for an negligible ϵ in the security parameter of the EMV system.

The experiment works as follows: attacker participate in the protocol for which it controls at least one entity involved, there are two Merchants, one gets paid with an associated transcript pay. The clearing is executed for the payment and finally the attacker guess the chosen Merchants based on its view.

Theorem 4

Assuming that the relation $\mathcal{R}^{\mathcal{P}_{\mathsf{cor}}}_{\mathcal{M}\mathtt{An}} \subsetneq \mathcal{M} \times \mathsf{PAY}$ is preimage resistant for a payment system and for a set of entities $\mathcal{P}_{\mathsf{cor}}$ then Game $\mathcal{M}\mathtt{An}$ is achieved for the corrupted set $\mathcal{P}_{\mathsf{cor}}$.

Proof. This proof is essentially the same as the first part of the proof of Theorem 3 on page 151.

Assume for contradiction that an adversary $\mathcal{A}$ can break $\mathsf{Exp}^{\mathcal{M}\mathtt{An}}_{\mathcal{P}_{\mathsf{cor}}}$ for a corruption set $\mathcal{P}_{\mathsf{cor}}$ while the relation $\mathcal{R}^{\mathcal{P}_{\mathsf{cor}}}_{\mathcal{M}\mathtt{An}}$ still achieves preimage resistance. Under these assumptions, the protocol in use achieves *Pseudonymity* but not *Game Based Merchant Pseudonymity*. Based on the adversary's interactions and its calls to the available oracles, two set $\mathcal{M} = \{M_0, M_1\}$ and $\mathsf{PAY}^{\mathcal{P}_{\mathsf{cor}}}$ can be defined, as well as a relation $\mathcal{R}^{\mathcal{P}_{\mathsf{cor}}}_{\mathcal{M}\mathtt{An}} \subsetneq \mathcal{M} \times \mathsf{PAY}^{\mathcal{P}_{\mathsf{cor}}}$.

In order to win against the game based Merchant pseudonymity, the adversary $\mathcal{A}$ returns the identity M_b of a payment pay_b. When $\mathcal{A}$'s answer is correct, assumed with non negligible probability, we have found a preimage for one of the elements in $\mathcal{R}^{\mathcal{P}_{\mathsf{cor}}}_{\mathcal{M}\mathtt{An}}$. Hence, based on this the adversarial algorithm we can construct a sequence of execution determining a relation $\mathcal{R}^{\mathcal{P}_{\mathsf{cor}}}_{\mathcal{M}\mathtt{An}}$ for which it is possible to recover the Merchant's identity. This contradicts our hypothesis and shows that preimage resistance of $\mathcal{R}^{\mathcal{P}_{\mathsf{cor}}}_{\mathcal{M}\mathtt{An}}$ implies security against $\mathsf{Exp}^{\mathcal{M}\mathtt{An}}_{\mathcal{P}_{\mathsf{cor}}}$. □

With this proof, we have reduced our mathematical relationship-based model to the game-based model presented above. This gives us two formalisms for the security of our proposals. In addition, because we have reduced the security of our relational model to

this model and prove the security of our proposals for the relationship-based model, our protocols meet the security requirements of both models.

5.14 Conclusion of the Chapter

The global EMV payment system is being modernised, and card usage is increasingly important in the global payment landscape. Consequently, cash payments are declining, and with it the privacy regarding most of our expenses is vanishing. Based on thorough investigations, we observed that most traditional payment methods lack measures of privacy for Payers and Merchants and their transactions. We have also discussed the predominant factors contributing to the privacy limitations in these methods, the most useful regulations on payment systems such as AML, KYC, PSD2 and SCA. We have formalised specific properties tailored for card payment scenarios. We introduced `PrivBank` and `PrivProxy`: two practical, law-abiding and fully EMV-compliant proposals designed to achieve payment pseudonymity, payment unlinkability, and Merchant pseudonymity against the relevant entities in the network. We have formalised these three properties and proved them for `PrivBank` and `PrivProxy`, on a new mathematical model for payments' privacy, as well as in the more traditional one.

Chapter 6

Conclusion

In this thesis, we have examined digital signatures and the EMV card payment protocol architecture to enhance anonymity in all these specific use cases. Nevertheless, we have concentrated on properties that implement only partial concepts of anonymity and this for practical aspects. Throughout our discussions, anonymity took on different meanings depending on the context. For linkable ring signatures, we were seeking non-disclosure of the identity of the signer, even though all the signer's signatures could be linked together. We thoroughly examined this property. To provide even greater privacy for the signer, who in this case acts as a proxy for another party, we explored k-Times Full Traceable Proxy Signatures. This scheme conceals the proxy's identity and ensures unlinkability of the proxy's signatures. However, to enable accountability in case of flooding, this anonymity is revoked if more than k signatures are produced. Building on this, we developed k-Times Full Traceable Sanitizable Signatures, which offer slight modifications. The anonymity properties remain the same, but the proxy is now only permitted to sanitize already signed messages, *i.e.,* modify parts of the message and update the signature accordingly.

Our third contribution, moves out from anonymous signature to focus on an already deployed standard, the ubiquitous EMV card payment protocol. We started by examining whether it was possible to transfer the cryptographic property of anonymity to this system, which is subject to numerous laws and regulations. There, we demonstrated that it is indeed law abbiding to obtain architectures providing weaker forms of anonymity and proposed two solutions. In these solution, anonymity took again another meaning, closer to the decentralisation of identity knowledge than cryptographic anonymity as ussally considered. Indeed, the bank and the payment proxy needed to be able to cooperate at any time to be able to recover full knowledge of the identity.

Throughout this work, we have considered anonymity from different viewpoints and reachs. Before us, (full) anonymity has been discussed extensively in the literature for years (*e.g.,* [Cha83a], and various schemes, such as ring signatures [RST01] with no anonymity constraints, and anonymous proxy signatures [FP08], have been designed. However, we opted to temper this strong notion of anonymity, focusing instead on partial forms, as they can be usefull in certain practical cryptographic designs. Furthermore, as observed for EMV [emv22a], regulations and existing standards sometimes impose restrictions, as auditability and full awareness of user behaviour are often required.

Outcomes and Future Pathways. To discuss these topics, we began by introducing the subject at a high level. Chapter 1 outlined the motivations for our work and provided an overview of related general research. Following this, in Chapter 2, we presented the cryptographic foundations necessary for defining security models and constructing new cryptographic primitives. This structured approach enabled us to explore the three contributions of this manuscript and draw conclusions independently.

In Chapter 3, we explored anonymity models for linkable ring signatures and revealed a critical discrepancy between theoretical models and practical expectations. The original and most used models failed to guarantee the anonymity of users. This demonstrates that formalising the security model of even not so complex cryptographic primitives, is not an easy task. Our research demonstrated this by exhibiting counter-examples that leaked the identities of the signers despite achieving proven security. However, the model we identified as more accurate for formalising the property of anonymity had already been introduced but has not been reflected in subsequent work. While significant, unconsidered vulnerabilities in the use of these signatures may still be discovered, we ruled out the possibility of global insecurity in existing schemes by closely examining the structure of their proofs. In fact, our investigation has led us to identify a pattern, suggesting that hybrid arguments could potentially be applied to prove anonymity from already proven one-time anonymity. This raises open questions as to whether the weaker notion, one-time anonymity, could be generically strengthened into anonymity by considering an additional structural property of these signatures.

In Chapter 4, we explored the practical application of anonymous delegations for up to k instances using specific signature schemes. We introduced the concept of k-times fully traceable anonymity for both proxy signatures and sanitizable signatures, defining it within a newly proposed security model tailored to each case, instantiated by efficient schemes (with key and signature sizes logarithmic in k). Their security model was derived from both existing k-times fully traceable signatures, merged with models from proxy and sanitizable signature. Their security has been proven under the said model. In our considerations, we left has further work to consider if a generic instantiation for k-times fully traceable anonymity could potentially stem from our constructions.

In Chapter 5, we addressed some privacy concerns in EMV payments. EMV payments are currently trackable, by EMV readers, observers *etc.*, based on identifying metadata transmitted to all entities in the system. We identified significant gaps in user anonymity and payment unlinkability that could be closed by proxying transactions. Current systems, constrained by AML, KYC, and SCA regulations, always compromised privacy, as demonstrated by a survey of existing payment means. Our proposed protocols, two new payment architectures, `PrivBank` and `PrivProxy`, directly build up based on the EMV protocol, offer privacy-enhancing solutions. The thorough evaluation of these protocols demonstrated their feasibility and effectiveness in enhancing privacy without conflicting with regulatory requirements or existing norms. The advantages of these constructions are twofold: on the one hand, they demonstrate the feasibility of anonymous payment and, on the other, they highlight practical ways of obtaining it. Valuable consideration would be to look at other ways of improving anonymity by modifying the current standard in order to strengthen privacy protection in future revi-

sions, particularly when payment systems protocols will be revised to take into account post-quantum security.

Opening Discussion. All these works have considered anonymity from a cryptographic point of view in the computational model. The formalisation of privacy in our payment protocols also makes the link between the high-level definition of anonymity and cryptographic security experiments. Thus, by demonstrating the former, we were able to conclude that our work was secure in the latter. A number of related research avenues have emerged.

The complexity of formalising a security experience that corresponds to simple but general privacy expectations has been shown from Chapter 3. And in Chapter 4, formalising multiple properties, including k-times full traceability for proxy and sanitizable signature, have appear to be a non-trivial task. Formulating security is a complex task, which is why providing proof against specific and wide ranges of attacks remains a challenging. Based on these observations, one may wish to provide, if possible, a more user-friendly framework for the computational model to better formalise security and improve the confidence in the security guarantees of new primitives.

Furthermore, while Chapters 3 and 4 only address the cryptographic point of view of the three primitives. By the works of Chapter 5, we have seen the realist difference between idealised anonymity for cryptographic primitives and anonymity not being put in used, and confronted against laws, regulations and norms. Anonymity is not always a goal when building up new protocols *e.g.,* EMVCo has introduced the PAR to vanish any unlinkability in its protocols[1]. Indeed here it is a threshold of lobbies that sometime prevents from introducting improved privacy preserving properties. In the end, anonymity realisation (or non-realisation) seems to depends more on metadata thant cryptography when it has already been taken into account before. To elaborate a bit more, many cryptographic protocol can be made fully anonymous while some metadata may be lightly added to the protocol and remove some if not all of it. There is also the question of whether it is legal to deploy, under the current laws, other large-scale measures to preserve anonymity, such as in the TLS protocol or its equivalents. Advancing anonymity or privacy features without sacrificing functionality remains a challenging task, but it must continue to be pursued in order to ensure robust protection in all aspects.

[1] `https://www.securetechalliance.org/wp-content/uploads/EMVCo-PAR-WP-FINAL-April-2018.pdf`

Bibliography

[ACDMT05] Giuseppe Ateniese, Daniel H Chou, Breno De Medeiros, and Gene Tsudik. Sanitizable signatures. In *Computer Security–ESORICS 2005: 10th European Symposium on Research in Computer Security*, 2005.

[Acta] UK Public General Acts. The money laundering and terrorist financing (amendment) regulations 2019. `https://www.legislation.gov.uk/uksi/2019/1511/contents/made`, Last accessed on 08-30-24.

[Actb] UK Public General Acts. Proceeds of crime act 2002. `https://www.legislation.gov.uk/ukpga/2002/29/contents`, Last accessed on 08-30-24.

[AHAN+22] Diego F Aranha, Mathias Hall-Andersen, Anca Nitulescu, Elena Pagnin, and Sophia Yakoubov. Count me in! extendability for threshold ring signatures. In *IACR International Conference on Public-Key Cryptography*, 2022.

[AHKLOA23] Frederic A. Hayek, Mirko Koscina, Pascal Lafourcade, and Charles Olivier-Anclin. Generic privacy preserving private permissioned blockchains. In *Proceedings of the 38th ACM/SIGAPP Symposium on Applied Computing*, 2023.

[Ama23] Amazon. Shop with amazon pay, 2023. `https://pay.amazon.co.uk/using-amazon-pay`, Last accessed on 11-16-23.

[App23] Apple. Set up apple pay, 2023. `https://support.apple.com/en-us/HT204506`, Last accessed on 11-16-23.

[ASY06] Man Ho Au, Willy Susilo, and Siu-Ming Yiu. Event-oriented k-times revocable-iff-linked group signatures. In *ACISP 2006*, 2006.

[AT17] Nicholas Akinyokun and Vanessa Teague. Security and privacy implications of nfc-enabled contactless payment systems. In *Proceedings of the 12th international conference on availability, reliability and security*, 2017.

[ATSS+18] Wilson Abel Alberto Torres, Ron Steinfeld, Amin Sakzad, Joseph K Liu, Veronika Kuchta, Nandita Bhattacharjee, Man Ho Au, and Jacob Cheng. Post-quantum one-time linkable ring signature and application to ring

confidential transactions in blockchain (lattice ringct v1. 0). In *Information Security and Privacy: 23rd Australasian Conference, ACISP*, 2018.

[Aut] Financial Conduct Authority. Financial crime guide:a firm's guide to-countering financialcrime risks (fcg). `https://www.handbook.fca.org.uk/handbook/FCG.pdf`, Last accessed on 09-02-24.

[Ban21] European Central Bank. Central Bank Digital Currency: functional scope, pricing and controls, 2021. `https://www.ecb.europa.eu/pub/pdf/scpops/ecb.op286~9d472374ea.en.pdf`, Last accessed on 11-16-23.

[BB21] Angèle Bossuat and Xavier Bultel. Unlinkable and invisible γ-sanitizable signatures. In *International Conference on Applied Cryptography and Network Security*, 2021.

[BBC+23] Rohann Bella, Xavier Bultel, Céline Chevalier, Pascal Lafourcade, and Charles Olivier-Anclin. Practical construction for secure trick-taking games even with cards set aside. In *International Conference on Financial Cryptography and Data Security*, 2023.

[BBG+22] Danai Balla, Pourandokht Behrouz, Panagiotis Grontas, Aris Pagourtzis, Marianna Spyrakou, and Giannis Vrettos. Designated-verifier linkable ring signatures with unconditional anonymity. In *International Conference on Algebraic Informatics*, 2022.

[BBW+23] Ksenia Budykho, Ioana Boureanu, Stephan Wesemeyer, Daniel Romero, Matt Lewis, Yogaratnam Rahulan, Fortunat Rajaona, and Steve Schneider. Fine-grained trackability in protocol executions. In *30th Annual Network and Distributed System Security Symposium, NDSS 2023, San Diego, California, USA, February 27 - March 3, 2023*. The Internet Society, 2023.

[BCC+24] Ioana Boureanu, Liqun Chen, Tom Chothia, Pascal Lafourcade, Chris Newton, and Charles Olivier-Anclin. Emv-compliant & usable anonymity for contactless payments. In *Under submission*, 2024.

[BCDD20] Ioana Boureanu, Tom Chothia, Alexandre Debant, and Stéphanie Delaune. Security analysis and implementation of relay-resistant contactless payments. In *Proceedings of the 2020 ACM SIGSAC conference on computer and communications security*, 2020.

[BCI20] Ioana Boureanu, Liqun Chen, and Sam Ivey. Provable-security model for strong proximity-based attacks: With application to contactless payments. In *Proceedings of the 15th ACM Asia Conference on Computer and Communications Security*, 2020.

[BCM+14] Mike Bond, Omar Choudary, Steven J Murdoch, Sergei Skorobogatov, and Ross Anderson. Chip and skim: cloning emv cards with the pre-play attack. In *2014 IEEE Symposium on Security and Privacy*. IEEE, 2014.

[BDGS16] Jeffrey Burdges, Florian Dold, Christian Grothoff, and Marcello Stanisci. Enabling secure web payments with gnu taler. In *Security, Privacy, and Applied Cryptography Engineering: 6th International Conference, SPACE 2016, Hyderabad, India, December 14-18, 2016, Proceedings 6*. Springer, 2016.

[BDH+19] Michael Backes, Nico Döttling, Lucjan Hanzlik, Kamil Kluczniak, and Jonas Schneider. Ring signatures: logarithmic-size, no setup—from standard assumptions. In *Advances in Cryptology–EUROCRYPT 2019: 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19–23, 2019, Proceedings, Part III 38*, pages 281–311. Springer, 2019.

[BDK+18] Joppe Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, John M Schanck, Peter Schwabe, Gregor Seiler, and Damien Stehlé. Crystals-kyber: a cca-secure module-lattice-based kem. In *2018 IEEE European Symposium on Security and Privacy*, 2018.

[BDK24] Jan Bobolz, Jesus Diaz, and Markulf Kohlweiss. Foundations of anonymous signatures: Formal definitions, simplified requirements, and a construction based on general assumptions. *Cryptology ePrint Archive*, 2024.

[BEHM22] Jonathan Bootle, Kaoutar Elkhiyaoui, Julia Hesse, and Yacov Manevich. Dualdory: Logarithmic-verifier linkable ring signatures through preprocessing. In *European Symposium on Research in Computer Security*, 2022.

[BFF+09] Christina Brzuska, Marc Fischlin, Tobias Freudenreich, Anja Lehmann, Marcus xpage, Jakob Schelbert, Dominique Schröder, and Florian Volk. Security of sanitizable signatures revisited. In *Public Key Cryptography–PKC 2009: 12th International Conference on Practice and Theory in Public Key Cryptography*, 2009.

[BFLS10] Christina Brzuska, Marc Fischlin, Anja Lehmann, and Dominique Schröder. Unlinkability of sanitizable signatures. In *PKC 2010*, 2010.

[BH18] Xavier Boyen and Thomas Haines. Forward-secure linkable ring signatures. In *Information Security and Privacy: 23rd Australasian Conference, ACISP*, 2018.

[BHMY23] Sergiu Bursuc, Ross Horne, Sjouke Mauw, and Semen Yurkov. Provably unlinkable smart card-based payments. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, 2023.

[BKM06] Adam Bender, Jonathan Katz, and Ruggero Morselli. Ring signatures: Stronger definitions, and constructions without random oracles. In *Theory of Cryptography Conference*, 2006.

[BKP20] Ward Beullens, Shuichi Katsumata, and Federico Pintore. Calamari and falafl: logarithmic (linkable) ring signatures from isogenies and lattices.

In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 464–492. Springer, 2020.

[BL16] Xavier Bultel and Pascal Lafourcade. k-times full traceable ring signature. In *2016 11th International Conference on Availability, Reliability and Security (ARES)*, 2016.

[BLL+19] Xavier Bultel, Pascal Lafourcade, Russell WF Lai, Giulio Malavolta, Dominique Schröder, and Sri Aravinda Krishnan Thyagarajan. Efficient invisible and unlinkable sanitizable signatures. In *22nd IACR International Conference on Practice and Theory of Public-Key Cryptography*, 2019.

[BLO18] Carsten Baum, Huang Lin, and Sabine Oechsner. Towards practical lattice-based one-time linkable ring signatures. In *International Conference on Information and Communications Security*, pages 303–322. Springer, 2018.

[BLOR21] Xavier Bultel, Pascal Lafourcade, Charles Olivier-Anclin, and Léo Robert. Generic construction for identity-based proxy blind signature. In *14th International Symposium In Foundations and Practice of Security*, 2021.

[BM18] Pedro Branco and Paulo Mateus. A code-based linkable ring signature scheme. In *Provable Security: 12th International Conference, ProvSec 2018*, 2018.

[BMW03] Mihir Bellare, Daniele Micciancio, and Bogdan Warinschi. Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions. In *Advances in Cryptology—EUROCRYPT 2003: International Conference on the Theory and Applications of Cryptographic Techniques*, 2003.

[BOA24a] Xavier Bultel and Charles Olivier-Anclin. On the anonymity of linkable ring signatures. In *Proceedings of the 23rd International Conference on Cryptology And Network Security*, 2024.

[BOA24b] Xavier Bultel and Charles Olivier-Anclin. Taming delegations in anonymous signatures: k-times anonymity for proxy and sanitizable signature. In *Proceedings of the 23rd International Conference on Cryptology And Network Security*, 2024.

[BPS14] Christina Brzuska, Henrich C Pöhls, and Kai Samelin. Efficient and perfectly unlinkable sanitizable signatures without group signatures. In *Public Key Infrastructures, Services and Applications: 10th European Workshop*, 2014.

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *Proceedings of the 1st ACM*

Conference on Computer and Communications Security, pages 62–73, 1993.

[BSTP21] David Basin, Ralf Sasse, and Jorge Toro-Pozo. The emv standard: Break, fix, verify. In *2021 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2021.

[BSWW13] Christina Brzuska, Nigel P Smart, Bogdan Warinschi, and Gaven J Watson. An analysis of the emv channel establishment protocol. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, 2013.

[CDK+17] Jan Camenisch, David Derler, Stephan Krenn, Henrich C. Pöhls, Kai Samelin, and Daniel Slamanig. Chameleon-hashes with ephemeral trapdoors. In *PKC 2017*, 2017.

[CDS94] Ronald Cramer, Ivan Damgård, and Berry Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In *CRYPTO '94*, 1994.

[CFF+17] Véronique Cortier, Alicia Filipiak, Jan Florent, Said Gharout, and Jacques Traoré. Designing and proving an emv-compliant payment protocol for mobile devices. In *2017 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE, 2017.

[CGH04] Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited. *Journal of the ACM (JACM)*, 51(4):557–594, 2004.

[CH91] David Chaum and Eugène van Heyst. Group signatures. In *Workshop on the Theory and Application of Cryptographic Techniques*, 1991.

[Cha83a] David Chaum. Blind signatures for untraceable payments. In *Advances in Cryptology: Proceedings of Crypto 82*, 1983.

[Cha83b] David Chaum. Blind signatures for untraceable payments. In David Chaum, Ronald L. Rivest, and Alan T. Sherman, editors, *Advances in Cryptology*, Boston, MA, 1983. Springer US.

[CJ10] Sébastien Canard and Amandine Jambert. On extended sanitizable signature schemes. In *Cryptographers' Track at the RSA Conference*, 2010.

[Cou17] Geoffroy Couteau. *Zero-knowledge proofs for secure computation.* PhD thesis, Université Paris sciences et lettres, 2017.

[CP93] David Chaum and Torben Pryds Pedersen. Wallet databases with observers. In *Advances in Cryptology — CRYPTO' 92*, 1993.

[CS97a] Jan Camenisch and Markus Stadler. Efficient group signature schemes for large groups. In *Advances in Cryptology — CRYPTO*, 1997.

[CS97b] Jan Camenisch and Markus Stadler. Efficient group signature schemes for large groups. In *Advances in Cryptology — CRYPTO '97*, 1997.

[Cur23] Curve. Curve: Rule your money, 2023. `https://www.curve.com/en-gb/`, Last accessed on 11-16-23.

[Dam87] Ivan Bjerre Damgård. Collision free hash functions and public key signature schemes. In *Workshop on the Theory and Application of of Cryptographic Techniques*, pages 203–216. Springer, 1987.

[DH76] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Trans. Inf. Theory*, 1976.

[DL21] Jesus Diaz and Anja Lehmann. Group signatures with user-controlled and sequential linkability. In *IACR International Conference on Public-Key Cryptography*, 2021.

[Dol19] Florian Dold. *The GNU Taler system: practical and provably secure electronic payments*. PhD thesis, Université Rennes 1, 2019.

[DRP12] Joeri De Ruiter and Erik Poll. Formal analysis of the emv protocol suite. In *Theory of Security and Applications: Joint Workshop, TOSCA 2011, Saarbrücken, Germany, March 31-April 1, 2011, Revised Selected Papers*. Springer, 2012.

[DY83] Danny Dolev and Andrew Yao. On the security of public key protocols. *IEEE Transactions on information theory*, 1983.

[eBa23] eBay. ebay payment terms of use, 2023. `https://pages.ebay.co.uk/payment/2.0/terms.html`, Last accessed on 11-16-23.

[eca22] The ECASH Act, 2022. `https://ecashact.us/`, Last accessed on 11-16-23.

[Ede23] Edenred, 2023. `https://www.edenred.fr/ticket-restaurant`, Last accessed on 02-24-24.

[EKCGD14] Ali El Kaafarani, Liqun Chen, Essam Ghadafi, and James Davenport. Attribute-based signatures with user-controlled linkability. In *Cryptology and Network Security: 13th International Conference, CANS 2014*, 2014.

[EKG17] Ali El Kaafarani and Essam Ghadafi. Attribute-based signatures with user-controlled linkability without random oracles. In *Cryptography and Coding: 16th IMA International Conference, IMACC 2017*, 2017.

[ElG85] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE transactions on information theory*, 1985.

[EMV11] LLC EMVCo. Emv integrated circuit card specifications for payment systems, book 2, security and key management, version 4.3, 2011.

[EMV14] LLC EMVCo. Emv next generation. next generation kernel system architecture overview., 2014.

[EMV21] LLC EMVCo. Emv payment tokenisation specification – technical framework. *EMVCo: Foster City, CA, USA*, 2021.

[emv22a] Overview of EMVCo, 2022. `https://www.emvco.com/about/overview/`, Last accessed on 11-16-23.

[EMV22b] LLC EMVCo. Emv payment tokenisation frequently asked questions (faq) — general. `https://www.emvco.com/specifications/emv-payment-tokenisation-faqs/`, 2022.

[EMV22c] LLC EMVCo. Emv payment tokenisation specification technical framework v2.3. *EMVCo: Foster City, CA, USA*, 2022.

[EMV23a] LLC EMVCo. Emv annual report 2023: Enhancing emv technologies to supporting emerging payments. *EMVCo: Foster City, CA, USA*, 2023.

[EMV23b] LLC EMVCo. Emv payment tokenisation – a guide to use cases. *EMVCo: Foster City, CA, USA*, 2023.

[EU16] EUR-Lex European Union. Regulation on the protection of natural persons with regard to the processing of personal data and on the free movement of such data. `https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A32016R0679`, Last accessed on 05-30-24, 2016.

[EU18] EUR-Lex European Union. Regulatory technical standards for strong customer authentication (sca) and common and secure open standards of communication. `https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A02018R0389-20230912`, Last accessed on 02-21-24, 2018.

[EU21a] EUR-Lex European Union. Directive on payment services in the internal market. `https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A02015L2366-20151223`, Last accessed on 02-21-24, 2021.

[EU21b] EUR-Lex European Union. Directive on the prevention of the use of the financial system for the purposes of money laundering or terrorist financing. `https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A02015L0849-20210630`, Last accessed on 02-21-24, 2021.

[FGK+22] Dario Fiore, Lydia Garms, Dimitris Kolonelos, Claudio Soriente, and Ida Tucker. Ring signatures with user-controlled linkability. In *European Symposium on Research in Computer Security*, 2022.

[FGL21] Ashley Fraser, Lydia Garms, and Anja Lehmann. Selectively linkable group signatures—stronger security and preserved verifiability. In *International Conference on Cryptology and Network Security*, 2021.

[FHS19] Georg Fuchsbauer, Christian Hanser, and Daniel Slamanig. Structure-preserving signatures on equivalence classes and constant-size anonymous credentials. *Journal of Cryptology*, 2019.

[FKM+16] Nils Fleischhacker, Johannes Krupp, Giulio Malavolta, Jonas Schneider, Dominique Schröder, and Mark Simkin. Efficient unlinkable sanitizable signatures from signatures with re-randomizable keys. In *19th IACR International Conference on Practice and Theory in Public-Key Cryptography*, 2016.

[FKP16] Manuel Fersch, Eike Kiltz, and Bertram Poettering. On the provable security of (ec) dsa signatures. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 1651–1662, 2016.

[FM21] Marc Fischlin and Arno Mittelbach. An overview of the hybrid argument. *Cryptology ePrint Archive*, 2021.

[FP08] Georg Fuchsbauer and David Pointcheval. Anonymous proxy signatures. In *Security and Cryptography for Networks: 6th International Conference*, 2008.

[FS87] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *Advances in Cryptology — CRYPTO*, 1987.

[FS07a] Eiichiro Fujisaki and Koutarou Suzuki. Traceable ring signature. In *International Workshop on Public Key Cryptography*, 2007.

[FS07b] Eiichiro Fujisaki and Koutarou Suzuki. Traceable ring signature. In *Public Key Cryptography – PKC 2007*, 2007.

[GL19] Lydia Garms and Anja Lehmann. Group signatures with selective linkability. In *Public-Key Cryptography–PKC 2019: 22nd IACR International Conference on Practice and Theory of Public-Key Cryptography*, 2019.

[GM84] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 1984.

[GM17] Jens Groth and Mary Maller. Snarky signatures: Minimal signatures of knowledge from simulation-extractable snarks. In *Annual International Cryptology Conference*, 2017.

[GMR89] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on computing*, 1989.

[GMR19] Shafi Goldwasser, Silvio Micali, and Chales Rackoff. The knowledge complexity of interactive proof-systems. In *Providing sound foundations for cryptography: On the work of shafi goldwasser and silvio micali*, pages 203–225. 2019.

[gnu22] Cental bank accounts are dangerous and unnecessary, a critique of two papers, 2022. https://taler.net/papers/accounts-dangerous-2022.pdf, Last accessed on 11-16-23.

[Gol01] Oded Goldreich. *Foundations of Cryptography, Volume 1*. Cambridge university press Cambridge, 2001.

[GR04] Craig Gentry and Zulfikar Ramzan. Eliminating random permutation oracles in the even-mansour cipher. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 32–47. Springer, 2004.

[GY19] L.-A. Galloway and T. Yunusov. First contact: New vulnerabilities in contactless payments. In *Black Hat Europe*, 2019.

[Han11] Jocelyn Hanamirian. The right to remain anonymous: Anonymous speakers, confidential sources and the public good. *Colum. JL & Arts*, 35:119, 2011.

[HC24] Xiangyu Hui and Sid Chi-Kin Chau. Llring: Logarithmic linkable ring signatures with transparent setup. In Joaquin Garcia-Alfaro, Rafał Kozik, Michał Choraś, and Sokratis Katsikas, editors, *Computer Security – ESORICS 2024*. Springer Nature Switzerland, 2024.

[HMY22] Ross Horne, Sjouke Mauw, and Semen Yurkov. Unlinkability of an improved key agreement protocol for emv 2nd gen payments. In *2022 IEEE 35th Computer Security Foundations Symposium (CSF)*. IEEE, 2022.

[HVDH21] David Harvey and Joris Van Der Hoeven. Integer multiplication in time o(n log(n)). *Annals of Mathematics*, 193(2):563–617, 2021.

[Inc23] Mastercard Inc. Mastercard prepaid | just load and pay, 2023. https://www.mastercard.co.uk/en-gb/personal/find-a-card/general-prepaid-mastercard.html, Last accessed on 11-16-23.

[Inc24] Mastercard Inc. Mastercard payment account reference inquiry, 2024. https://developer.mastercard.com/payment-account-reference-inquiry/documentation/, Last accessed on 08-29-24.

[KL06] Marek Klonowski and Anna Lauks. Extended sanitizable signatures. In *Proceedings of the 9th International Conference on Information Security and Cryptology*, ICISC'06, 2006.

[KLM+22] Mirko Koscina, Pascal Lafourcade, Gaël Marcadet, Charles Olivier-Anclin, and Léo Robert. A survey on identity-based blind signature. In *15th International Symposium In Foundations and Practice of Security*, 2022.

[KM15] Neal Koblitz and Alfred J Menezes. The random oracle model: a twenty-year retrospective. *Designs, Codes and Cryptography*, 77:587–610, 2015.

[KSS15] Stephan Krenn, Kai Samelin, and Dieter Sommer. Stronger security for sanitizable signatures. In *International Workshop on Data Privacy Management*, 2015.

[LASZ13] Joseph K Liu, Man Ho Au, Willy Susilo, and Jianying Zhou. Linkable ring signature with unconditional anonymity. *IEEE Transactions on Knowledge and Data Engineering*, 2013.

[LAZ19] Xingye Lu, Man Ho Au, and Zhenfei Zhang. Raptor: a practical lattice-based (linkable) ring signature. In *Applied Cryptography and Network Security: 17th International Conference, ACNS 2019, Bogota, Colombia, June 5–7, 2019, Proceedings 17*, pages 110–130. Springer, 2019.

[leg11] legislation.gov.uk. The electronic money regulations 2011, 2011. `https://www.legislation.gov.uk/uksi/2011/99/2016-03-01`, Last accessed on 11-16-23.

[LMMOA24] Pascal Lafourcade, Dhekra Mahmoud, Gael Marcadet, and Charles Olivier-Anclin. Transferable, auditable and anonymous ticketing protocol. In *Proceedings of the 19th ACM ASIA Conference on Computer and Communications Security*, 2024.

[LMOAR24] Pascal Lafourcade, Lola-Baie Mallordy, Charles Olivier-Anclin, and Léo Robert. Secure keyless multi-party storage scheme. In *Proceedings of the 29th European Symposium on Research in Computer Security*, 2024.

[LNY+19] Zhen Liu, Khoa Nguyen, Guomin Yang, Huaxiong Wang, and Duncan S Wong. A lattice-based linkable ring signature supporting stealth addresses. In *Computer Security–ESORICS 2019: 24th European Symposium on Research in Computer Security*, 2019.

[LW05] Joseph K Liu and Duncan S Wong. Linkable ring signatures: Security models and new schemes. In *International Conference on Computational Science and Its Applications–ICCSA 2005*, 2005.

[LWW04] Joseph K Liu, Victor K Wei, and Duncan S Wong. Linkable spontaneous anonymous group signature for ad hoc groups. In *Information Security and Privacy: 9th Australasian Conference, ACISP*, 2004.

[Lyf23] Lyf, 2023. `https://www.lyf.eu/en/`, Last accessed on 02-23-24.

[LYMW13] Weiwei Liu, Guomin Yang, Yi Mu, and Jiannan Wei. k-time proxy signature: Formal definition and efficient construction. In *Provable Security: 7th International Conference, ProvSec 2013, Melaka, Malaysia, October 23-25, 2013. Proceedings 7*, 2013.

[Mau05] Ueli Maurer. Abstract models of computation in cryptography. In *Cryptography and Coding: 10th IMA International Conference, Cirencester, UK, December 19-21, 2005. Proceedings 10*, pages 1–12. Springer, 2005.

[MDAB10] Steven J Murdoch, Saar Drimer, Ross Anderson, and Mike Bond. Chip and pin is broken. In *2010 IEEE Symposium on Security and Privacy*. IEEE, 2010.

[MUO96] Masahiro Mambo, Keisuke Usuda, and Eiji Okamoto. Proxy signatures for delegating signing operation. In *Proceedings of the 3rd ACM Conference on Computer and Communications Security*, CCS '96, 1996.

[NB08] Satoshi Nakamoto and A Bitcoin. A peer-to-peer electronic cash system. *Bitcoin.–URL: https://bitcoin. org/bitcoin. pdf*, 2008.

[Pay22] PayPal. Paypal: Account limits, 2022. `https://www.paypal.com/uk/smarthelp/article/faq1253`, Last accessed on 17-05-22.

[Pay23] Google Pay. Google pay help: United kingdom: Supported payment methods, 2023. `https://support.google.com/pay/answer/7351534`, Last accessed on 11-16-23.

[Ped91] Torben Pryds Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In *Annual international cryptology conference*, 1991.

[PS00] David Pointcheval and Jacques Stern. Security arguments for digital signatures and blind signatures. *Journal of cryptology*, 2000.

[PS19] Sunoo Park and Adam Sealfon. It wasn't me! In *Annual International Cryptology Conference*, 2019.

[Raz02] Rosehaslina Razali. Overview of e-cash: Implementation and security issues, 2002. `https://www.giac.org/paper/gsec/1799/overview-e-cash-implementation-security-issues/103204`, Last accessed on 11-16-23.

[RCN+22] Andreea-Ina Radu, Tom Chothia, CJ Newton, Ioana Boureanu, and Liqun Chen. Practical emv relay protection. In *Proc. 43rd IEEE Symp. Security Privacy*, 2022.

[Rev23] Revolut. Spend safely online with single-use cards, 2023. `https://www.revolut.com/cards`, Last accessed on 11-16-23.

[RSA78] Ronald L Rivest, Adi Shamir, and Leonard Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[RST01] Ronald L Rivest, Adi Shamir, and Yael Tauman. How to leak a secret. In *International conference on the theory and application of cryptology and information security*, 2001.

[Sch91] Claus-Peter Schnorr. Efficient signature generation by smart cards. *Journal of cryptology*, 1991.

[Sch24] Sven Schäge. New limits of provable security and applications to elgamal encryption. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 255–285. Springer, 2024.

[Sha79] Adi Shamir. How to share a secret. *Communications of the ACM*, 1979.

[Sho04] Victor Shoup. Sequences of games: a tool for taming complexity in security proofs. Cryptology ePrint Archive, Report 2004/332, 2004. https://eprint.iacr.org/2004/332.

[SLZ20] Eva-Maria Schomakers, Chantal Lidynia, and Martina Ziefle. All of me? users' preferences for privacy-preserving data markets and the importance of anonymity. *Electronic Markets*, 30, 02 2020.

[Swi23] Swidch. Otac: A new paradigm for user authentication and device authentication, 2023. https://www.swidch.com/technology/otac, Last accessed on 11-16-23.

[Tak24] Takepayments. Takepayments: What are contactless payments and how do they work?, 2024. https://www.takepayments.com/blog/product-information/what-are-contactless-payments-and-how-do-they-work/, Last accessed on 08-29-24.

[TFS04] Isamu Teranishi, Jun Furukawa, and Kazue Sako. k-times anonymous authentication (extended abstract). In *ASIACRYPT 2004*, 2004.

[TW05] Patrick P Tsang and Victor K Wei. Short linkable ring signatures for e-voting, e-cash and attestation. In *International Conference on Information Security Practice and Experience*, 2005.

[TWC+05] Patrick P Tsang, Victor K Wei, Tony K Chan, Man Ho Au, Joseph K Liu, and Duncan S Wong. Separable linkable threshold ring signatures. In *Progress in Cryptology-INDOCRYPT 2004: 5th International Conference on Cryptology in India*, 2005.

[TY21] Aleksei Stennikov Timur Yunusov, Artem Ivachev. New vulnerabilities in public transport schemes for apple pay, samsung pay, gpay. *White Paper*, 2021.

[UO23] Up-One, 2023. https://up-one.up.coop, Last accessed on 02-24-24.

[Val24] Valimised. Internet voting in estonia, 2024. https://www.valimised.ee/en/internet-voting-estonia, last access 06/19/2024.

[Vis21] Visa. Minimum data requirements for merchants, 2021. https://www.elavon.ie/content/dam/elavon/en-gb/documents/customer-centre/customer-news/merchant-best-practice-infographic.pdf, Last accessed on 02-21-24.

[Vis22] Visa. Visa prepaid reloadable personal cards, 2022. https://www.visa.co.uk/content/VISA/usa/englishlanguagemaster/en_US/home/pay-with-visa/cards/prepaid-cards/all-purpose-reloadable.html, Last accessed on 01-06-22.

[Vis24] Visa. Visa scan to pay, 2024. https://developer.visa.com/innovation-corner/example-projects/scan-and-pay, Last accessed on 02-27-24.

[VS13] Nicolas Van Saberhagen. Cryptonote v 2.0. 2013.

[WSI02] Yodai Watanabe, Junji Shikata, and Hideki Imai. Equivalence between semantic security and indistinguishability against chosen ciphertext attacks. In *Public Key Cryptography—PKC 2003: 6th International Workshop on Practice and Theory in Public Key Cryptography Miami, FL, USA, January 6–8, 2003 Proceedings 6*, pages 71–84. Springer, 2002.

[WYM14] Jiannan Wei, Guomin Yang, and Yi Mu. Anonymous proxy signature with restricted traceability. In *2014 IEEE 13th International Conference on Trust, Security and Privacy in Computing and Communications*, 2014.

[WYML15] Jiannan Wei, Guomin Yang, Yi Mu, and Kaitai Liang. Anonymous Proxy Signature with Hierarchical Traceability. *The Computer Journal*, 2015.

[XLAZ24] Yuxi Xue, Xingye Lu, Man Ho Au, and Chengru Zhang. Efficient linkable ring signatures: New framework and post-quantum instantiations. In Joaquin Garcia-Alfaro, Rafał Kozik, Michał Choraś, and Sokratis Katsikas, editors, *Computer Security – ESORICS 2024*. Springer Nature Switzerland, 2024.

[YLA+13] Tsz Hon Yuen, Joseph K Liu, Man Ho Au, Willy Susilo, and Jianying Zhou. Efficient linkable and/or threshold ring signature without random oracles. *The Computer Journal*, 2013.

[Yun21] Timur Yunusov. Hand in your pocket without you noticing: Current state of mobile wallet security. *Black Hat Europe*, 2021.

[ZLL+19] Lingyue Zhang, Huilin Li, Yannan Li, Yong Yu, Man Ho Au, and Baocang Wang. An efficient linkable group signature for payer tracing in anonymous cryptocurrencies. *Future Generation Computer Systems*, 2019.

[ZLS+20] Xinyu Zhang, Joseph K Liu, Ron Steinfeld, Veronika Kuchta, and Jiangshan Yu. Revocable and linkable ring signature. In *Information Security and Cryptology: 15th International Conference, Inscrypt 2019*, 2020.

Appendix A

Résumé Long

Avec l'avènement de la cryptographie à clef publique et son évolution, la notion cryptographique de *privacy*[1] a pris des sens multiples. Il ne s'agit plus comme dans un premier temps de garantir uniquement la confidentialité des messages, *i.e.,* la non-divulgation des messages claire. La non divulgation de l'identité des utilisateurs ou des dispositifs destinataires *i.e.,* le respect de la *vie privé* s'est aussi retrouvé au centre de l'attention.

Pour s'assurer de l'identité d'une entité numérique, un mécanisme cryptographique permettant l'authentification a été développé: la *signature électronique*. Le premier *schéma de signature électronique*, assurant l'authenticité et l'intégrité des échanges fût introduite par Rivest, Shamir et Adelman [RSA78]. Ce méchanisme est basé sur une paire de clefs, et est le pendant direct des signatures manuscrites que nous connaissons tous. Pour fournir plus d'intuition sur le concept, nous l'illustrons dans la Figure A.1 en représentant la signature comme un sceau (apposé sur une lettre), qui fût une ancienne manière de singer. Le point rouge représente la signature associé à une clef publique qui donne des signatures "rouges". Cette couleur est considérées comme uniques, *i.e.,* toute autre clef publique donnerait une signature d'une couleur différente. Les signatures électroniques peuvent être utilisées pour vérifier l'authenticité et l'intégrité des messages ou documents digitales, garantissant que l'identité de l'expéditeur est confirmée et que le contenu n'a pas été altéré pendant la transmission ou le stockage. Elles doivent être infalsifiables, *i.e.,* elles ne peuvent pas être produites facilement sans connaître la clef secrète liée à la clef publique.

Figure A.1: Sceau d'une signature électronique[2].

Aujourd'hui, les signatures électroniques jouent un rôle crucial dans la protection contre les fraudes, les falsifications et les violations de données. Elles permettent également d'authentifier le propriétaire d'une clef publique sous certaines hypothèses de confiance, déportant ainsi notre confiance a un faible nombre d'entités. Elle permet donc des comuncation authentifiés plus faciles et plus sûres.

[1] Les traductions françaises seraientt dans ce contexte *confidentialité* ou *vie privé*. Elles donnent la part belle à cette multiplicité de senses.

[2] Ces représentations schématiques et celles à venir sont inspirées de la présentation de [AHAN+22] par Elena Pagnin lors de PKC '22.

Cependant, la mise en œuvre de mécanismes d'authentification soulève d'importantes questions de confidentialité des échanges. En effet, les identités sont aujourd'hui considérées comme des informations sensibles. Leur usage c'est répendu malgré ce fait que dans de nombreux cas d'applications, il ne soit pas nécessaire d'authentifier un utilisateur spécifique. Prenons l'exemple d'un document en lecture seule partagé par plusieurs personnes et stocké sur un serveur. Est-il nécessaire d'authentifier la personne accédant au document, ou est-il suffisant d'empêcher tout accès non autorisé ? Dans de nombreux cas, le document n'est pas sensible, empêcher l'accès non autorisé peut donc suffire. Dans ces circonstances, il est suffisant de prouver le droit d'accès au document, sans pour autant révéler l'identité de l'utilisateur. Bien que cela puisse sembler surprenant, cela peut être réalisé grâce à l'utilisation de techniques cryptographiques classiques comme des schémas de signatures électronique.

Ce type de mécanisme d'authentification préservant la *privacy* est précisément ce que nous explorons dans cette thèse. Ils relèvent du domaine des primitives cryptographiques asymétriques modernes et nous nous concentrons spécifiquement sur l' *anonymat* des entités. Nos principaux objectifs et contributions impliquent l'étude des méthodes d'authentification basées sur les signatures, la formalisation de leur sécurité et le développement de moyens pratiques et efficaces pour protéger l'identité des personnes. Pour garantir l'applicabilité et la praticité de nos primitives, nous prenons en compte la nécessité de limiter l'anonymat dans de nombreux des domaines d'application. L'anonymat total, peut conduire à des fraudes ou à des comportements répréhensibles qui restent impunis. Par conséquent, nous considérons la nécessité d'exiger la *liabilité* des signatures électronique, la *traçabilité* de l'utilisateur en cas de fraude ou même dans certains cas l'*auditabilité* complète du système, entre autres.

A.1 Modèle d'Anonymat pour les Signature d'Anneaux Liables

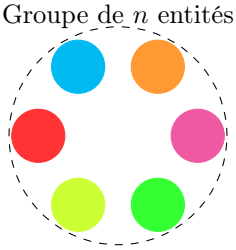


Figure A.2: Sceau d'une signature en anneau [RST01].

Introduites par Rivest, Shamir et Tauman [RST01] en 1986, les *signatures d'anneaux* sont des signatures électroniques qui permettent à une entité de signer au nom d'un groupe *ad hoc* tout en cachant son identité au vérificateur. Nous avons schématiquement décrit le sceau de ce type schéma de signature dans la Figure A.2, où plusieurs points représentent plusieurs entités à l'intérieur de l'anneau, mais le cercle les comprenant est en pointillé car l'anneau dans ce type de signature est *ad hoc*, c'est-à-dire qu'il est généré par le signataire au moment de la signature et n'a pas nécessité d'acceptation

de la part des autres membres inclus dans l'anneau (en dehors de la génération de leurs clefs publiques). Cela signifie qu'un seul des n signataires (dans la Figure A.2, $n = 6$) a effectivement signé le message, le sceau des autres membres du groupe ayant été en quelque sorte imité par le signataire. Les signatures d'anneaux partagent les mêmes propriétés que les signatures électroniques : elles doivent être difficiles à contrefaire pour une personne ne possédant pas la clef secrète. De plus, toute signature produite par un signataire est indistinguable de celles générées par d'autres membres du groupe, garantissant ainsi l'anonymat des signataires au sein du groupe. Nous avons examiné une primitive appelée *signatures d'anneaux liables*. Les signatures d'anneaux liables atténuent la propriété d'anonymat des signatures d'anneaux en liant les signatures émises par le même signataire. Il est attendu que deux signatures provenant de la même entité soient toujours liables, tandis que les signatures de différentes entités doivent rester non liées. Ces propriétés de la signature en anneaux liables sont complétées par celles des signatures d'anneaux. La propriété de liabilité est visuellement décrite dans la Figure A.3. Intuitivement cela equivaux à ce que chaque signataire laisse son empreinte digitale lors de la signature. Les empreinte digitale n'évoluant pas dans le temps, elles sont identiques lors de la signature de deux messages différents, mais diffèrent lorsque deux entités signent. Ici, nous avons simplement utilisé la couleur pour distinguer les empreintes digitales ; il n'y a aucun lien avec le signataire potentiel derrière la signature. Cette propriété trouve des cas d'applications pratiques, par exemple lorsque nous voulons déterminer si des demandes proviennent de la même entité et ce sans avoir besoin de connaître son identité.

Figure A.3: Sceau d'une signature en anneaux liables et leur liabilité [RST01].

Dans des travaux récents tels que [BEHM22] et dans tous les a autres schémas de signature d'anneaux liables existant (voir Figure A.3 et à l'exception de deux d'entre eux: [ATSS+18] et [BKP20]), l'*Anonymat* (ano) est informellement caractérisé par l'énoncé suivant:

> *"Anonymity, demands that an adversary cannot tell which of a ring's secret keys was used to produce a signature.*[3]*"*

Malgré cette description informelle précise, nous avons montré dans cette thèse que les définitions de tous les schémas listés dans le Tableau A.1a formalisent essentiellement ce même concept de la manière suivante :

> *L'anonymat exige qu'un adversaire ne puisse pas identifier quelle clef secrète de l'anneau a été utilisée pour produire* ***la première signature d'une entité****.*

[3]Traduction : « L'anonymat exige qu'un adversaire ne puisse pas déterminer laquelle des clefs secrètes de l'anneau a été utilisée pour produire une signature. »

Référence	Hypothèse	Modèle
Liu *et al.* [LWW04]	Liée à DL	ROM
Tsang *et al.* [TWC+05]	RSA Fort & DDH	ROM
Liu et Wong [LW05]	Liée à DL	ROM
Tsang et Wei [TW05]	Liée à DL	ROM
Liu *et al.* [LASZ13]	Liée à DL	ROM
Yuen *et al.* [YLA+13]	Liée à DL	Standard
Boyen et Haines [BH18]	CDL	ROM
Branco et Mateus [BM18]	GSDD	ROM
Baum *et al.* [BLO18]	SIS, LWE	ROM
Lu *et al.* [LAZ19]	SIS	ROM
Liu *et al.* [LNY+19]	M-SIS, D-MLWE	ROM
Zhang *et al.* [ZLS+20]	Liée à DL	ROM
Balla *et al.* [BBG+22]	Liée à DL	ROM
Bootle *et al.* [BEHM22]	Liée à DL	ROM
Xiangyu *et al.* [HC24]	Liée à DL	ROM
Xue *et al.* [XLAZ24]	Construction générique	ROM

(a) Signatures en anneaux liables existantes prouvées sécurisées sous le modèle pour l'anonymat à usage unique 1-ano.

Référence	Hypothèse	Modèle
Alberto *et al.* [ATSS+18]	R-SIS	ROM

(b) Signatures en anneaux liables à usage unique existantes.

Référence	Hypothèse	Modèle
Backes *et al.* [BDH+19]	Construction générique	Standard
Beullens *et al.* [BKP20]	SIDH, M-LWE	ROM

(c) Signatures en anneaux liables existantes avec anonymat prouvé ano.

Table A.1: Signatures en anneaux liables existantes.

Nous voyons une implication directe du deuxième énoncé par le premier. Nous ferons référence à la deuxième citation et à cette notion plus faible sous le terme *anonymat unique* (1-ano).

Dans la pratique, ces deux énoncés ont des formulations relativement similaires, mais ils diffèrent grandement dans les garanties aux signatures qui les utilisent pour prouver leur sécurité.

Contributions

Ansi, nous avons démontré que la modélisation de l'expérience d'anonymat pour les signatures d'anneaux liables ne correspond pas aux attentes de sécurité formalisées par 16 des 18 schémas existants.

Le modèle de sécurité que nous avons nommé anonymat unique (1-ano), décrit dans la Figure A.4a, reste largement similaire dans tous les travaux existants. Il ne donne accès qu'à une seul signature produite par un signataire dont l'adversaire doit retrouver l'identité. Ainsi, cette formalisation ne garantie pas l'anonymat pour la deuxième signature produite à partir d'une paire de clefs et cela ne correspond pas aux attentes informelles décrites dans tous ces travaux. Notre contribution principale est de mettre en évidence l'absence de l'adoption d'un formalisme approprié pour l'anonymat, même dans certaines des recherches les plus récentes.

En effet, un autre modèle existe dans la littérature mais n'a été utilisé que pour 2 des 18 schémas existants [ATSS+18, BKP20]. Ce modèle prend mieux en compte l'anonymat attendu des signatures d'anneaux liables. Il est illustré dans la Figure A.4a. Nous montrons, par la contruction d'un schéma de signatures d'anneaux liables, que la propriété d'anonymat ano est strictement plus forte que la propriété d'anonymat unique 1-ano. Ce schéma contre-exemple se base sur un schéma de partage secret [Sha79] composé des algorithmes Split et Recover et n'importe quel schéma de signatures d'anneaux liables LRS existant. Le contre-exemple est comme suite:

$\mathsf{CeLRS.Setup}_{\mathsf{LRS}}(1^\lambda)$**:** correspond à l'exécution de $\mathsf{LRS.Setup}_{\mathsf{LRS}}(1^\lambda)$.

$\mathsf{CeLRS.Gen}_{\mathsf{LRS}}(1^\lambda)$**:** exécute $(\mathsf{sk}_{\mathsf{LRS}}, \mathsf{pk}_{\mathsf{LRS}}) \leftarrow \mathsf{LRS.Gen}_{\mathsf{LRS}}(1^\lambda)$ et $s_1, s_2 \leftarrow \mathsf{Split}(\mathsf{pk}_{\mathsf{LRS}}, 2)$. Définit et renvoie $\mathsf{sk} = (\mathsf{sk}_{\mathsf{LRS}}, s_1, s_2)$, $\mathsf{pk} = \mathsf{pk}_{\mathsf{LRS}}$.

$\mathsf{CeLRS.Sign}_{\mathsf{LRS}}(\mathsf{sk}_i, m, \{\mathsf{pk}_j\}_{j\in\mathcal{R}})$**:** décompose sk_i en $(\mathsf{sk}_{\mathsf{LRS}}, s_1, s_2)$, tire aléatoirement $b \xleftarrow{\$} \{1, 2\}$ et renvoie $\sigma_{\mathsf{LRS}} \leftarrow \mathsf{LRS.Sign}_{\mathsf{LRS}}(\mathsf{sk}_{\mathsf{LRS}}, m\|s_b, \{\mathsf{pk}_j\}_{j\in\mathcal{R}})$ et s_b comme σ.

$\mathsf{CeLRS.Verif}_{\mathsf{LRS}}(m, \sigma, \{\mathsf{pk}_j\}_{j\in\mathcal{R}})$**:** décompose σ en σ_{LRS} et s. Exécute $\mathsf{LRS.Verif}_{\mathsf{LRS}}(m\|s, \sigma_{\mathsf{LRS}}, \{\mathsf{pk}_j\}_{j\in\mathcal{R}})$ et renvoie le résultat.

$\mathsf{CeLRS.Link}_{\mathsf{LRS}}(\sigma, \sigma')$**:** décompose σ en σ_{LRS} et s, et σ' en σ'_{LRS} et s'. Exécute et renvoie le résultat de $\mathsf{LRS.Link}_{\mathsf{LRS}}(\sigma_{\mathsf{LRS}}, \sigma'_{\mathsf{LRS}})$.

D'autres contre-exemple se basant sur la littérature existante sont aussi présentés [BL16, ATSS+18]. Nous argumentons donc en faveur de la notion la plus forte: l'anonymat ano.

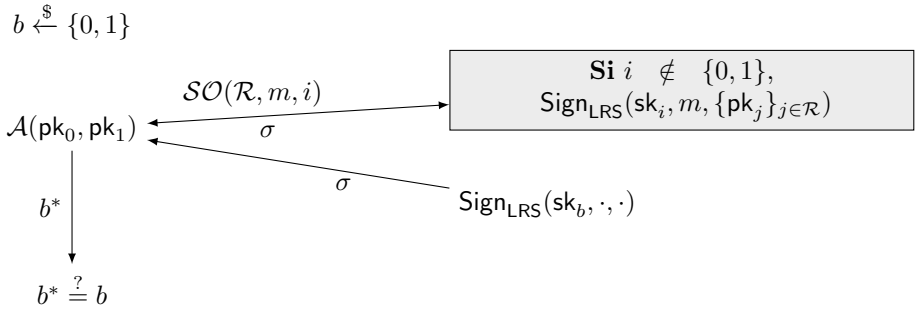
De plus, les signatures d'anneaux liables admettent deux modèles de corruption :

Le modèle des clefs honnêtes : un scénario d'attaque où toutes les clefs de signature doivent avoir été générées honnêtement par le challenger dans l'expérience.

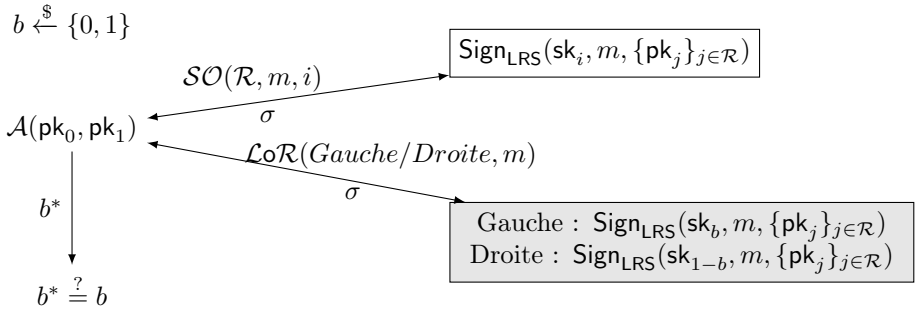
Le modèle des clefs choisies par l'adversaire : un scénario d'attaque où les clefs de signature peuvent avoir été générées de manière malveillante par l'adversaire.

Notre contribution s'étend à une classification complète des propriétés d'anonymat dans les deux modèles de corruption pour les signatures d'anneaux liables. Pour cela, un second contre-exemple est nécessaire pour démontrer la différence stricte entre les deux modèles de corruption. Nous le construisons sur la base d'un schéma de chiffrement IND-CPA et de l'un des schémas de signature en anneaux liables de la littérature.

Ensuite, nous passons en revue tous les travaux existants initialement basés sur la notion plus faible d'anonymat unique 1-ano. En s'intéressant aux preuves de schémas existants, nous avons observé que beaucoup suivent un schéma de preuve qui pourraient potentiellement être étendu par de simples arguments hybrides. Cependant, deux schéma existent déjà dans le modèle le plus fort [ATSS+18, BKP20]. Nous n'avons donc pas essayé de démontrer la sécurité des autres signatures existantes.



(a) Anonymat Unique 1-ano des signatures d'anneaux liables.



(b) Anonymat ano des signatures d'anneaux liables.

Figure A.4: Comparaison schématique des expériences d'anonymat pour les signatures d'anneaux liables. (Les modèles de corruption ne sont pas spécifiés.)

A.2 Signature Délégables et Assainissable k-fois parfaitement traçable

Introduites par Mambo, Usuda et Okamoto [MUO96] en 1996, les *signatures par délégué* permettent à une entité, le *signataire*, parfois appelé le *signataire original*, de déléguer ses droits de signature à une entité, appelée *proxy*, qui peut alors signer des documents en leur nom. Le raison d'être de cette primitive est de facilité la délégation sécurisée de signatures électroniques, comme par exemple pour donner une procuration pour une élection. Dans la Figure A.5, nous pouvons voir le sceau du signataire rouge donnant la délégation au signataire orange qui appose son propre sceau, la partie rouge agissant comme un certificat, une preuve de la délégation donnée. Le concept original a ensuite été étendu de différentes manières. Notre intérêt se concentre principalement sur la primitive introduite par Fuchsbauer et Pointcheval [FP08] appelée *signatures par délégué anonyme*. Ces signatures offrent l'anonymat au délégué tout en maintenant l'intégrité et l'authenticité du message signé. Ainsi, seul le nom du signataire original est divulgué.

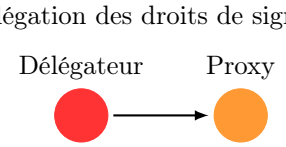
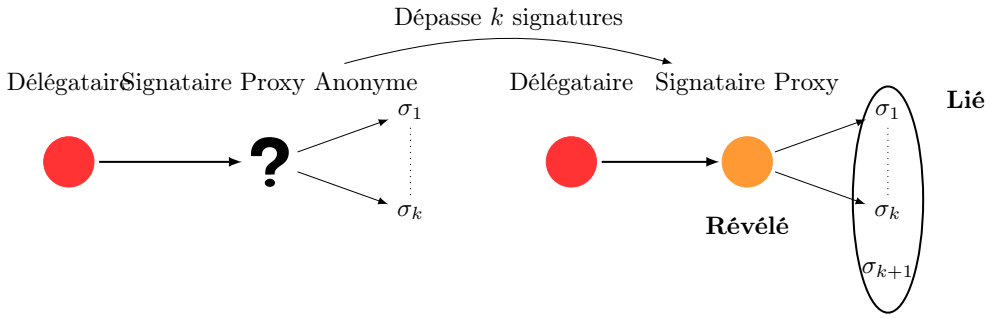
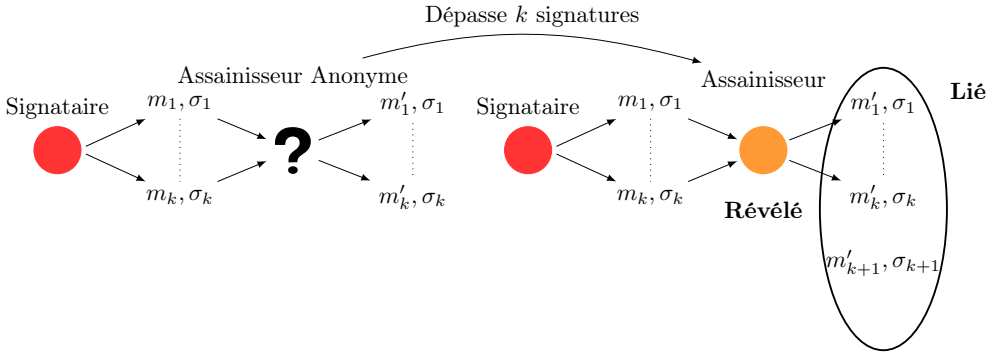


Figure A.5: Sceau des signatures par délégué.



(a) k-times Signature Proxy Anonyme Complètement Traçable.



(b) Signature Délégable et Assainissable k-fois Parfaitement traçable.

Figure A.6: Signature Proxy et Sanitizable Anonyme Complètement Traçable k-times.

Contributions

Pour limiter l'anonymat complet des signatures par délégué anonyme face au vérificateurs de signatures, nous avons introduit un nouveau type de schéma de signature anonymes avec délégations : les *signature délégable k-fois parfaitement traçable*. Dans ce type de signature, le délégué est anonyme, *i.e.,* il ne révèle pas son identité à moins qu'il n'émette plus de k signatures. Si la limite est dépassée, son identité est *révélée* et peut alors être *liée* à toutes les signatures qu'il a émises. Nous avons représenté visuellement ces signatures dans la Figure A.6a. Dans cette figure, nous voyons une délégation établie par le sceau rouge à une entité inconnue autorisée à émettre jusqu'à k signatures différentes. Si cette entité dépasse les k signatures autorisées en produisant une $k+1^{\text{ème}}$ signature, alors sa clef publique, associée à la couleur orange, est *révélée* et toutes les signatures produites sont *liées*. Ces signatures doivent atteindre trois propriétés en plus de la infalsifiabilité. Ces propriétés sont:

Anonymat : les signatures sont anonymes tant que le délégué ne dépasse pas la limite de k signatures. En particulier, elles ne peuvent pas être liées entre elles.

Traçabilité : si le délégué dépasse la limite de k signatures, il ne pourra pas empêcher que toutes ses signatures soient liées entre elles et que son identité soit retrouvée.

Innocuité : un délégué ne peut pas produire une signature qui pourrait être attribuée à une autre entité.

Ces propriétés ont été formalisés dans un modèle calculatoire.

La contruction de notre *signature délégable k-fois parfaitement traçable* nécessitent deux preuves à divulgation nulle de connaissance.

La preuve $\Pi_{<k}$ garantit que le délégué ne peut pas produire plus de k signatures. En pratique, elle assure que le prouveur connaît un élément s et un entier η tels que (i) $\widetilde{y}_i = y_{i,j}^s$ et $\widetilde{\mathsf{pk}}_i = \mathsf{pk}_{i,j}^s$ sont bien formés selon s, un entier η de l bits (avec $l = \log_2(k)$) et les valeurs publiques $y_{i,j}$ et $\mathsf{ppk}_{i,j}$, pour $i \in \{0, \ldots, k-1\}$ et $j \in \{0, 1\}$. Et (ii) $\eta < k$.

Prouver (i) revient à prouver $(\widetilde{g}_1 = g_1^s \wedge \widetilde{y}_i = y_{i,0}^s \wedge \widetilde{\mathsf{pk}}_i = \mathsf{pk}_{i,0}^s)$ ou $(\widetilde{g}_1 = g_1^s \wedge \widetilde{y}_i = y_{i,1}^s \wedge \widetilde{\mathsf{pk}}_i = \mathsf{pk}_{i,1}^s)$ pour tout $i \in [\![0, l]\!]$. Formulé avec la notation de Camenisch et Stadler [CS97a], cela donne :

$$\mathsf{ZK}\left\{ s : \bigwedge_{i=0}^{l} \bigvee_{j=0}^{1} \left(\widetilde{g}_1 = g_1^s \wedge \widetilde{y}_i = y_{i,j}^s \wedge \widetilde{\mathsf{pk}}_i = \mathsf{pk}_{i,j}^s \right) \right\}.$$

La transcription de cette preuve est linéaire en l. D'autre part, prouver (ii) consiste à prouver $\eta < k$, où chaque bit $\eta[i]$ de η est engagé dans l'égalité $\widetilde{y}_i = y_{i,\eta[i]}^s$. Ainsi, pour prouver que η est plus petit que k, nous devons comparer k et η en tant que mots binaires à travers les engagements $\widetilde{y}_i$, en parcourant les bits du plus significatif au moins significatif.

D'autre part, une seconde preuve à divulgation nulle de connaissance, appelée π_σ, est aussi utilisée pour lier tous les éléments de traçabilité et prouver la bonne connaissance du secret. Elle utilise des bases de construction similaires à celles de $\Pi_{<k}$ et est inspirée par la construction *signature d'anneaux k-fois parfaitement traçable* de [BL16]. Ici nous décrivons la preuve que nous avons utilisé avec des notations génériques (où pour tout entier i, chaque g_i, γ_i, et h_i est un élément d'un groupe $\mathbb{G}_i$ d'un même ordre premier p). La preuve π_σ est la suivante :

$$\pi_\sigma \leftarrow \mathsf{ZK}\left\{ x, y, z : \begin{array}{l} h_1 = {g_1}^x \wedge h_2 = {g_2}^y \wedge h_3 = {g_3}^x \wedge h_4 = {g_4}^z \\ h_5 = {g_5}^x \cdot \gamma_5^y \wedge h_6 = {g_6}^x \cdot \gamma_6^y \wedge h_7 = {g_7}^y \end{array} \right\}.$$

Notre construction exige que ces deux preuves à divulgation nulle de connaissance soient *consistantes*, *robustes*, *ne divulges aucune information* et sont *extractibles*. Nous nous sommes aussi basé sur une *signature sur classes d'équivalence* qui est *infalsifiable*, *cachant les classes*, et *adaptable* [FHS19], sur une *fonction de hachage résistante aux collisions*. A l'aide de tous ces éléments nous avons démontre que le schéma de *signature délégable k-fois parfaitement traçable* possède l'ensemble des propriétés de sécurité décrites plus haut sous l'hypothèse Décisionnelle de Diffie-Hellman DDH.

À partir de nos *signature delegable k-fois parfaitement traçable*, nous avons pu dériver une *signature assainissable k-fois parfaitement traçable*. Les signatures assainissable agissent d'une manière relativement similaire aux signatures par délégué, ajoutant le fait que les messages sont partiellement prescrits par le délégataire. Nous avons également représenté visuellement ces signatures, cette fois dans la Figure A.6b et introduisons le premier schéma de *signature assainissable k-fois parfaitement traçable*.

En plus des propriété d'infalsifiabilité, d'anonymat, de traçabilité et d'innocuité. Les propriétés de sécurité sont assurés par les propriétés de sécurité découlants des signatures assainissable :

Immutabilité : il n'est pas possible de modifier des parties des messages qui ne doivent pas être modifiées.

Transparence : il n'est pas possible de deviner si une signature a été assainie ou non. Cette propriété implique une autre propriété existante appelée *vie privée* : il n'est pas possible de déterminer des informations sur le message original.

Non-liabilité : il n'est pas possible de lier une signature assainie à la signature originale, ou de lier des signatures assainies provenant de la même signature originale. Quelques schémas tels que [FKM+16] atteignent cette propriété. Notez que la non-liabilité diffère de l'anonymat, qui garantit qu'il n'est pas possible de lier des signatures provenant du même utilisateur.

Invisibilité : il n'est pas possible d'identifier quelle partie du message est modifiable. Notez que concevoir des schémas qui sont à la fois non-lienables et invisibles est un défi, et il n'existe que deux schémas dans la littérature qui combinent ces propriétés [BLL+19, BB21].

Découlant de notre *signature delegable k-fois parfaitement traçable*, notre *signature assainissable k-fois parfaitement traçable* s'appuis sur les preuves à divulgation nulle de connaissance $\Pi_{<k}$ et π_σ afin d'instancier des *signatures par la connaissance* [GM17] ce qui est possible grace à l'heuristique de Fiat-Shamir [CS97b]. Les *signatures par la connaissance* ainsi obtenus obtiennent les propriétés de *consistantes*, *robustes*, *ne divulges aucune information* et sont *extractibles*. Encore une fois, nous nous sommes aussi basé sur une *signature sur classes d'équivalence* qui est *infalsifiable*, *cachant les classes*, et *adaptable* [FHS19], sur une *fonction de hachage résistante aux collisions*. A l'aide de tous ces éléments nous avons démontre que le schéma de *signature assainissable k-fois parfaitement traçable* possède l'ensemble des propriétés de sécurité décrites plus haut sous l'hypothèse Décisionnelle de Diffie-Hellman DDH.

A.3 Anonymat utilisable en conformité avec la norme EMV pour les paiements sans contact

Dans les contributions précédentes nous avons discuté de l'anonymat de signatures electroniques d'un point de vue cryptographique. Les systèmes numériques et de communication ont été normé depuis leur première apparition, et de nombreuses réglementations ont été introduites. Ces systèmes initialement conçus de toutes pièces suivent désormais des cadres largement adoptés. L'introduction de modifications nécessite le respect des aspects légaux et des normes existantes et doit offrir des améliorations par rapport aux systèmes existants pour être standardisé et, par la suite, être déployés. En soi, concevoir un nouveau protocole avec des fonctionnalités améliorées est un défi, mais lorsqu'il s'agit de l'adoption de normes et de déploiement, c'est presque impossible. Nous nous somme intéressé à un standard parmis d'autres: le standard de paiement par carte bancaire EMV (pour *Europay Mastercard Visa*). Le deuxième standard mondial pour les systèmes de paiement, visant à améliorer considérablement l'efficacité du protocoles actuels et à prévenir de nombreuses attaques, a été introduit en 2013. À l'heure actuelle, 11 ans plus tard, il n'a toujours pas remplacé la première version du protocole existant

depuis 1996, ce qui montre le temps et la volonté considérables nécessaires pour modifier des systèmes établis à grande échelle.

Alors que cette nouvelle norme attend son déploiement, diverses propositions [HMY22, BHMY23] ont été faites pour l'améliorer, anticipant ainsi des changements par rapport à ce qui a été proposé. Les autheurs de ces articles notent que l'anonymat a été négligé dans la norme de service de paiement actuellement utilisée et dans sa deuxième version. Ainsi dans les protocole actuel et le potentiel future protocole : toutes les informations liées à l'identité sont entièrement diffusées à toutes les entités participant au paiement, ou même à une entité qui écouterais les communications. Les auteurs de [HMY22, BHMY23] ont tenté d'aborder la non liabilité et l'anonymat face des entités qui écouteraient les communications entre la carte et le terminal de paiement. Ces propriété constituent un pas vers plus de confidentialité des paiements. Cependant, nous notons qu'il n'est pas nécessaire que toutes les informations soient connues de tous les partis lorsqu'un paiement est effectué ; comme le montrent les transactions en espèces.

L'absence d'anonymat pour les clients vis-à-vis des entités impliquées dans le système souligne la nécessité de poursuivre la recherche et les améliorations en matière de confidentialité des clients. Surtout alors que les préoccupations en matière de vie privé s'intensifient et que les droits à la vie privé deviennent plus formalisés. Il est donc crucial de veiller à ce que les futurs systèmes de paiements protègent efficacement l'anonymat des utilisateurs tout en répondant également aux exigences de détection de fraude.

Cependant, il est difficile de concilier les exigences légales avec la vie privée des utilisateur du système de paiements. Les régulations contre le blanchiment d'argent [EU21b] (*AML*), comme le *Know Your Customer (KYC)* (qui requière la connaissance de l'identité des clients par l'émetteur) et l'*Authentification Forte du Client* [EU18] (SCA), visent à protéger contre la fraude mais nuisent à l'instoration d'anonymat dans les paiements EMV, qui doivent être audités. Ainsi, un schéma de paiement respectant la vie privée doit rester conforme aux lois et régulations qui régissent les paiements.

Contributions

Notre objectif fût de construire un systeme de payment en magasin (*i.e.,* par carte ou mobile) compatible avec les lois, les régulations et la norme EMV en place. Ici nous proposons deux solutions à ce problème sous les noms de `PrivBank` et `PrivProxy`, elles se basent toutes deux sur un tiers parti apportant de l'anonymat à l'utilisateur. Afin de pouvoir discuter l'anonymat qu'apportent ces constructions, nous avons commencé par proposer un ensemble de notions de confidentialité souhaitables, suffisamment générales pour être pertinentes dans la plupart des systèmes de paiement. Ces propriétés peuvent être considérées du point de vue des différentes entités impliquées : face à un commerçant, un émetteur de cartes (*i.e.,* une banque) ou un intermédiaire. Nous proposons un ensemble de notions de confidentialité applicables à divers systèmes de paiement, en tenant compte des différentes entités impliquées.

Pseudonymat du Payeur ($\mathcal{P}$`An`). Une instance de $\mathcal{P}$`An` est respectée si un tiers ne peut pas connaître l'identité du payeur ID ou son pseudonyme à long terme. Deux variantes existent :

Méthode de paiement	SCA		KYC		Pseudonymat		
	Émetteur	Proxy	Émetteur	Proxy	Émetteur	Commercant	Proxy
1. Espèces	non	N.A.	N.A.	N.A.	$\mathcal{M}$An	$\mathcal{P}$An, Unlnk	N.A.
2. Chèque	y	N.A.	y	N.A.	$\neg\,\mathcal{M}$An	$\neg\,\mathcal{P}$An, $\neg$ Unlnk	N.A.
3. E-cash	y	N.A.	y	N.A.	$\mathcal{M}$An	$\mathcal{P}$An, Unlnk	N.A.
4. Cartes physiques	y	N.A.	y	N.A.	$\neg\,\mathcal{M}$An	$\neg\,\mathcal{P}$An, $\neg$ Unlnk	N.A.
5. Google, Apple Pay, *etc.*	y	N.A.	y	N.A.	$\neg\,\mathcal{M}$An	$\mathcal{P}$An, $\neg$ Unlnk	N.A.
6a. Cartes prépayées	y	N.A.	y	N.A.	$\neg\,\mathcal{M}$An	$\mathcal{P}$An/$\neg\,\mathcal{P}$An, $\neg$ Unlnk	$\neg\,\mathcal{P}$An, $\neg$ Unlnk, $\neg\,\mathcal{M}$An
6b. Cartes-cadeaux	non	(n)	n	(n)	$\mathcal{M}$An/$\neg\,\mathcal{M}$An	$\mathcal{P}$An, Unlnk/$\neg$ Unlnk	($\neg\,\mathcal{P}$An, $\neg$ Unlnk, $\neg\,\mathcal{M}$An)
7. Cartes virtuelles	y	(y)	y	(y)	$\mathcal{M}$An/$\neg\,\mathcal{M}$An	$\mathcal{P}$An, Unlnk	($\neg\,\mathcal{P}$An, $\neg$ Unlnk, $\neg\,\mathcal{M}$An)
8a. PayPal	y	y/non	y	y/non	$\mathcal{M}$An	$\mathcal{P}$An, $\neg$ Unlnk	$\neg\,\mathcal{P}$An, $\neg$ Unlnk, $\neg\,\mathcal{M}$An
8b. Curve	y	y	y	y	$\neg\,\mathcal{M}$An	$\neg\,\mathcal{P}$An, $\neg$ Unlnk	$\neg\,\mathcal{P}$An, $\neg$ Unlnk, $\neg\,\mathcal{M}$An
9. Marketplaces en ligne	y	non	y	y/non	$\mathcal{M}$An	$\mathcal{P}$An, $\neg$ Unlnk	$\neg\,\mathcal{P}$An, $\neg$ Unlnk, $\neg\,\mathcal{M}$An
PrivBank	y	non	y	non	$\mathcal{M}$An	$\mathcal{P}$An, Unlnk	$\mathcal{P}$An, $\neg$ Unlnk, $\neg\,\mathcal{M}$An
PrivProxy	y	y	non	y	$\mathcal{M}$An	$\mathcal{P}$An, Unlnk	$\neg\,\mathcal{P}$An, $\neg$ Unlnk, $\neg\,\mathcal{M}$An

Table A.2: Propriétés SCA, KYC et pseudonymat des méthodes de paiement du point de vue de l'émetteur, du commerçant et du proxy lorsque celui-ci existe. $\neg\,\mathcal{P}$An et $\neg$ Unlnk s'appliquent à tous les systèmes pour l'émetteur, et $\neg\,\mathcal{M}$An à tous les systèmes pour le commercant. Des explications détaillées sont fournies dans la Section 5.6. *(Notation : N.A. signifie "non applicable", $\mathcal{P}$An pour $\mathcal{P}\mathit{An}^{ID}$ et $\mathcal{P}\mathit{An}^{C_{ID}}$, les crochets indiquent que le proxy n'existe pas nécessairement dans tous les systèmes, et le / signifie que le déploiement peut entraîner des propriétés différentes.)*

$\mathcal{P}$An^{ID} **:** le commerçant ne découvre pas l'identité à long terme du payeur.

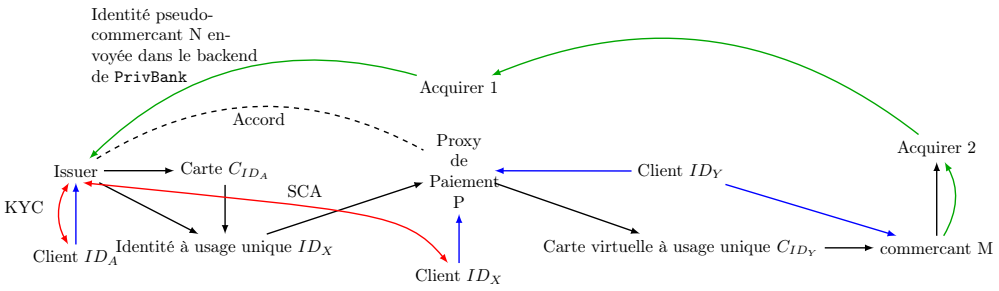
$\mathcal{P}$An$^{C_{ID}}$ **:** le commerçant ne découvre pas le numéro de carte à long terme du payeur.

Désassociabilité des Paiements (Unlnk). Une instance de Unlnk est respectée si aucune entité ne peut lier des paiements effectués par le même payeur. Par exemple, les paiements en espèces dans un même magasin, par une même entité, ne peuvent pas être reliés.

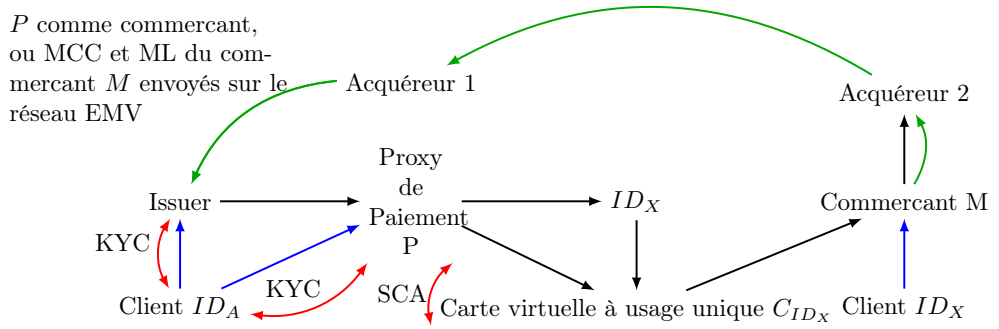
Pseudonymat du Commerçant ($\mathcal{M}$An). Une instance de $\mathcal{M}$An est respectée si une entité (comme un émetteur) ne peut pas identifier le commerçant impliqué dans un paiement. Actuellement, des informations sur le commerçant sont transmises aux banques, ce qui rend leur identification possible.

Nous avons commencé par examiné ce que l'on pourrait appeler les *"systèmes de paiement traditionnels"* ainsi que certains systèmes *modernes*[4], bien qu'ils soient construits sur les précédents. Nous avons aussi discuté de leur adhérence aux régulations KYC et SCA, ainsi que de leur conformité à nos exigences de confidentialité $\mathcal{P}$An, Unlnk, et $\mathcal{M}$An. Nous avons exclu les crypto-monnaies [NB08] et les paiements basés sur des codes QR [Lyf23]. La raison en est que leur infrastructure est totalement différente de celle des paiements bien établis, en particulier les systèmes basés sur EMV, que nous visons à améliorer ici. Ainsi, les systèmes d'intérêt dans notre analyse étaient : 1. l'argent liquide, 2. le chèque, 3. l'e-cash [Cha83b], 4. les cartes bancaires physiques/-classiques, 5. les applications mobiles [App23, Pay23], 6. les cartes rechargeables et prépayées [Inc23, Vis22], 7. les cartes virtuelles/à usage unique [Rev23], 8. les prestataires de services de paiement tels que 8a. PayPal [Pay22] et 8b. Curve [Cur23], et 9. les places de marché en ligne [Ama23, eBa23]. Nous pensons que le lecteur est familier avec la plupart de ces systèmes au point de pouvoir juger s'ils respectaient les normes SCA, KYC, $\mathcal{P}$An, Unlnk, $\mathcal{M}$An. Les résultats sont compilés dans le Tableau A.2.

[4]Les paiements en ligne sont inclus ici car ils peuvent fournir des indications sur ce qui pourrait être possible.



(a) `PrivBank`: Paiements conformes à l'EMV avec pseudonymat provisionné collaborativement par un émetteur respectueux de la vie privée et un proxy tiers.



(b) `PrivProxy`: Paiements conformes à l'EMV avec pseudonymat provisionné par un proxy tiers.

Figure A.7: Description graphique de nos deux propositions. (Les flèches noires indiquent le flux d'exécution. Les flèches rouges représentent les exigences KYC et SCA. Les flèches bleues indiquent la connaissance d'identité. Les flèches vertes représentent les demandes de compensation.)

`PrivBank` et `PrivProxy`. Nous proposons deux constructions compatibles avec les paiements sans contact EMV, offrant les propriétés de confidentialité $\mathcal{P}$`An`, $\mathcal{M}$`An`, `Unlnk`, avec des garanties prouvables, tout en respectant les lois et régulations (SCA/PSD2, AML, *etc.*).

Dans notre première construction, appelée `PrivBank` et illustrée schématiquement par la Figure A.7a, une banque émettrice respectueuse de la vie privée fournit les propriétés d'anonymat $\mathcal{P}$`An` et `Unlnk` à ses clients, en partenariat avec une entité Payment Proxy qui gère les paiements et garantie la propriété d'anonymat $\mathcal{M}$`An`.

Dans la seconde construction, appelée `PrivProxy` et illustrée schématiquement par la Figure A.7b, le rôle de la banque est remplacé par une entité Payment Proxy respectueux du pseudonymat, qui offre les propriétés d'anonymat $\mathcal{P}$`An`, $\mathcal{M}$`An` et `Unlnk`, tandis que les payeurs choisissent indépendamment leur banque. Cependant les propriétés d'anonymat de ce système sont moins fortes.

Ces constructions composent des paiements standards, *i.e.,* basé sur EMV et non privés, ou leurs composants, afin d'obtenir un paiement EMV mobile sans contact, anonyme (avec $\mathcal{P}$`An`, $\mathcal{M}$`An`, `Unlnk` face à une partie des entités). Nous réalisons cela grâce à l'utilisation de proxys, sans modifier les éléments EMV des paiements originaux ni ajouter de cryptographie. Ainsi, la cryptographie utilisée dans nos schémas et les blocs de construction EMV sont traités comme des boîtes noires héritées d'EMV. Nos

preuves pour les propriétés d'anonymats $\mathcal{P}$`An`, $\mathcal{M}$`An`, `Unlnk` découlent de la construction par un intermédiaire, indépendamment des détails cryptographiques.

Afin d'étudier et d'apporter des garanties de sécurité pour les constructions proposés, nous avons introduit un formalisme qui a permis de raisonner formellement sur nos propositions `PrivBank` et `PrivProxy`. Nous avons aussi formalisé les notions d'anonymat $\mathcal{P}$`An`, `Unlnk` et $\mathcal{M}$`An`. Ce formalisme a été conçu pour être accessible à un large public et se basse sur des relations logiques. Par ailleurs, nous avons également fourni un modèle cryptographique traditionnel, basé sur des experiences de sécurité, et une analyse de nos schémas. Nous avons démontré que les définitions simplifiées que nous avons présentés capturent pleinement les définitions cryptographiques.

A.4 Conclusion

Dans cette thèse, nous avons exploré les propriétés des signatures numériques et le protocole de paiement EMV pour améliorer l'anonymat dans des cas spécifiques. Nous nous sommes concentrés sur des formes partielles d'anonymat pour des raisons pratiques. Par exemple, pour les signatures en anneaux liables, nous avons étudié la non-divulgation de l'identité du signataire tout en permettant le lien entre ses signatures. Nous avons également proposé des signature délégables et assainissable k-fois parfaitement traçable pour préserver l'anonymat tout en permettant une responsabilité en cas d'abus. En ce qui concerne le protocole EMV, nous avons démontré que des formes faibles d'anonymat peuvent être respectées malgré les contraintes réglementaires. L'anonymat, dans ce contexte, dépend davantage des métadonnées que de la cryptographie. Nous pouvons conclure que l'amélioration des fonctionnalités de confidentialité tout en maintenant la conformité aux régulations est un défi, mais que le développement d'outils pour l'anonymat doit être poursuivie pour garantir une meilleure protection de la vie privée.